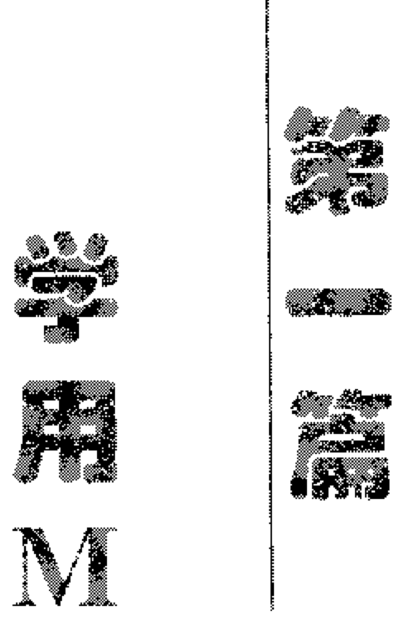
目录

第一篇　学用 MATLAB

第二篇 电路分析

第三篇 信号与系统分析



第一篇

学用MATLAB



第一章 MATLAB 简介

MATLAB 语言是一种广泛应用于工程计算及数值分析领域的新型高级语言，其顶尖的数值计算功能、强大的图形可视化功能、简洁易学的便笺式的编程语言、可交互的集成环境，深受工程技术人员的欢迎。本章介绍 MATLAB 6.x 的集成环境、管理指令和帮助系统。

1.1 MATLAB 特点

1. 数值计算功能

在当前的科学计算中，几乎无处不用矩阵运算，而 MATLAB 具有丰富的矩阵运算函数和简单的指令格式。

在 MATLAB 中，每个数值元素都视为复数，而且只有双精度(64 位)一种数据格式，省去多种数据格式的设置，虽然在运行速度和内存消耗方面付出了代价，却使 MATLAB 的编程大大简化，这正是 MATLAB 的主要目标。虽然数据格式只有一种，但为了人机交互的方便，输出显示格式有 10 种。

MATLAB 的数值计算基本功能包括：矩阵运算、多项式和有理分式计算、数据统计分析以及数值积分等。

2. 符号计算功能

在实际应用中，除了数值计算外，还需要得到方程的解析解、简化和展开多项式和函数表达式、求解函数值等，所有这些均属于符号计算的领域。利用符号计算功能，可以直接进行公式的推导，求出方程的解析表达式。

3. 便笺式的编程语言

与 Fortran 和 C 等高级语言相比，MATLAB 的语法规则更简单，更贴近人的思维方式和表达习惯，使得编写程序就像在便笺上列公式和演算一样。

MATLAB 的表达式与数学、工程计算中常用的形式十分相似。在 MATLAB 中，若 A 为二维矩阵，b 和 x 是向量，矩阵 A 与 x 相乘被写成 A * x；如果给出方程 A * x = b，需要求解 x，只需键入 x = A\b 或 x = b/A 即可。由此可见，MATLAB

的语言规则与科技人员的书写习惯相近,易写易读易交流,不需强记编程规则。此外,矩阵的行列数也无需事先定义。

4. 强大而简易的作图功能

能根据输入数据自动确定坐标绘图。可绘制连续曲线、离散曲线、直方图、阶梯图和枝干图等。

可使用多种坐标系,例如直角坐标系、极坐标系,对数及半对数坐标系等。

能绘制三维坐标系的曲线、曲面和等高线,还可绘制四维图形。而且通常只需一条指令。例如,如果 a 是一个矩阵,则只需键入 plot(a)即可。

可在一个坐标系下绘制多条曲线,也可在一个图形窗口内绘制若干个独立的坐标系图形。

可分别用编程方式和交互方式对图形进行修饰,例如设置颜色、线型、加文字标注、三维图形的视角等。

5. 高智能化

绘图时自动选择最佳坐标,大大方便了用户。

作数值积分时按精度要求自动选择积分步长。

自动检测和显示程序错误,减轻编程和调试的工作量。

6. 丰富实用的工具箱

MATLAB 软件包括基本部分和专业扩展部分。

基本部分包括矩阵的运算和各种变换、方程求解、数据处理等,可以满足大学理工科类学生的需要。MATLAB 的核心内容在于它的基本部分,所有的工具箱子程序都是用基本部分的语句编写的,电路、信号与系统的分析也如此。

扩展部分称为工具箱。工具箱分为两类:功能性工具箱和学科性工具箱。功能性工具箱主要用来扩充其符号计算功能、可视建模仿真功能及文字处理功能等。学科性工具箱专业性比较强,如控制系统工具箱、信号处理工具箱、神经网络工具箱、最优化工具箱、金融工具箱、小波工具箱等。

1.2 MATLAB 6.x 的集成环境

在 Windows 桌面,双击 MATLAB 图标,系统就会进入如图 1-1 所示的 MATLAB 6.x 的工作环境。

MATLAB 6.x 的集成环境由桌面平台以及组件组成。它包含 8 个组成部分:指令窗口、历史指令窗口、工作台及工具箱窗口、当前工作目录窗口、工作空间窗口、矩阵编辑器、程序编辑器和帮助浏览器。

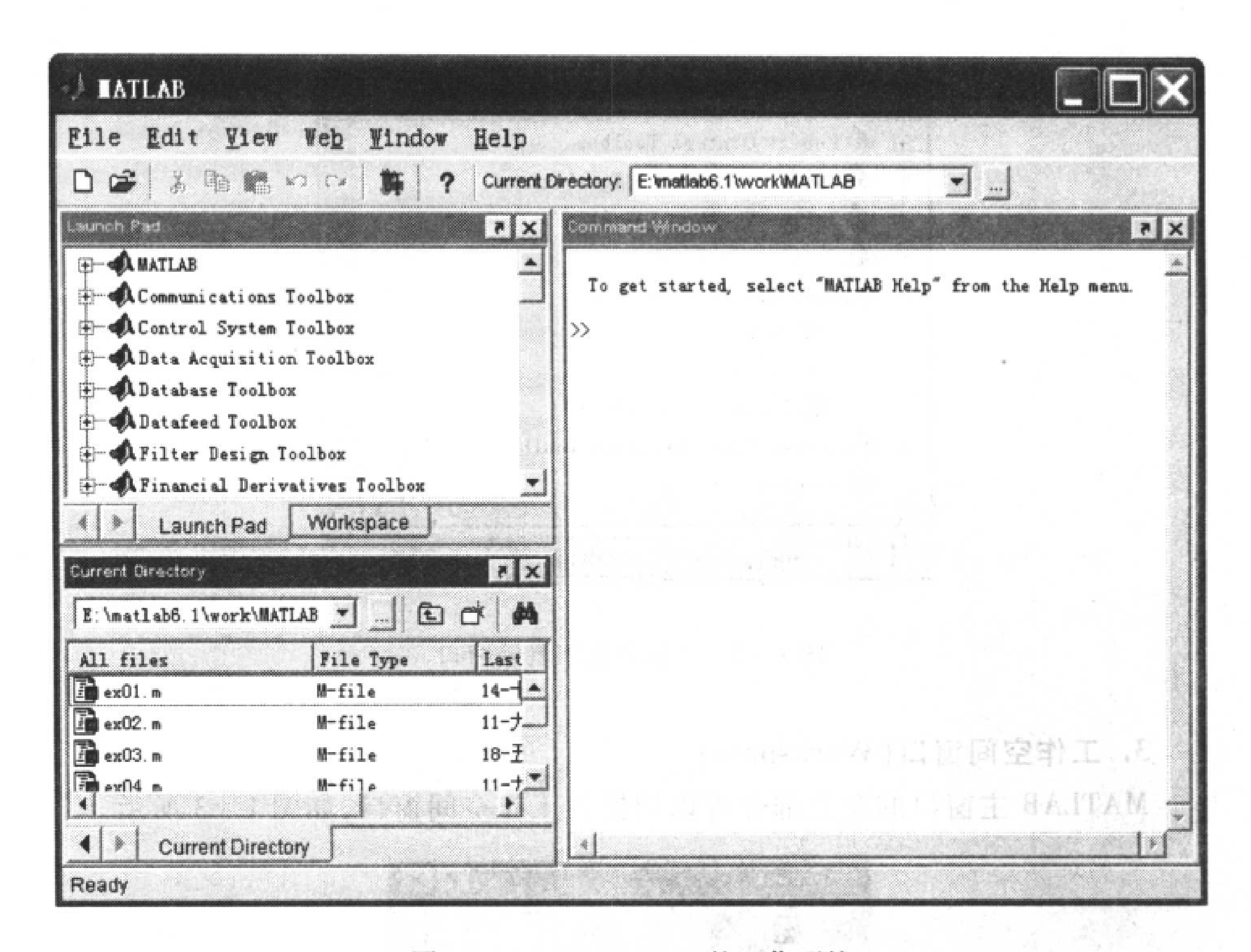


图 1-1 MATLAB 6.x 的工作环境

1. 指令窗口(Command Window)

在图 1-1 所示的右边空白部分,是 MATLAB 的指令窗口。指令窗口是 MATLAB 极为重要的部分,也是用户使用最频繁的部分。用户的数据输入和运算结果显示,一般都在此窗口中进行。

2. 工作台和工具箱窗口(Launch Pad)

在图 1-1 所示主窗口的左上部是 MATLAB 的工作台及工具箱窗口。在 MATLAB 6.x 的工作台及工具箱窗口中,可以看到已经安装的各种工具箱,双击选中的工具箱或单击前面的"+"号,就能看到工具箱中的各项功能。例如,单击 Symbolic Math Toolbox 前面的"+"号,可以看到这个工具箱中包含 Help、Demos和 Product Page(Web)三个部分,如图 1-2 所示。它们的作用分别如下:

Help:提供在线帮助。

Demos:提供系统演示功能。

Product Page(Web):提供 Internet 在线支持。

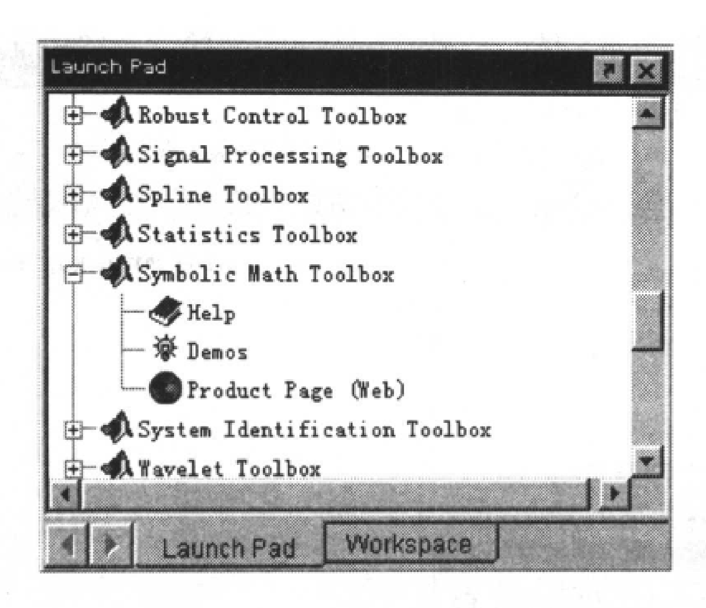


图 1-2 工作台和工具箱窗口

3. 工作空间窗口(Workspace)

MATLAB 主窗口的左上部分可以切换为工作空间窗口,如图 1-3 所示。



图 1-3 工作空间窗口

在工作空间窗口中,可以看到 MATLAB 的各个工作变量。其中 a、b 和 c 是用户在指令窗口(或在程序编辑器窗口)输入并运行的变量,ans 是一个由系统提供的默认输出变量。

在这里利用上方的一排工具栏提供的按钮,可以导入以前曾存入磁盘中的变量,也可以对选中的变量进行存盘、修改和删除操作。

需要注意的是，工作空间里的变量只是驻留在内存中，若想将矩阵的数据长期保留下来以备以后使用，就必须使用 MAT 文件对矩阵数据进行保存，在需要时再装载到工作空间中。例如，选中矩阵 a 后，单击上方的图标(Save)，把它以 MAT 后缀文件保存在磁盘中。使用时，单击上方的图标(Open)，再把它装载进来，也可以把它调入当前工作目录窗口中后进行双击操作。

4. 矩阵(数组)编辑器(Array Editor)

若要修改工作空间中的变量，只需双击此变量，即可打开图 1-4 所示的矩阵编辑器窗口。在这里可以改变矩阵的规模和元素的大小及表达式。

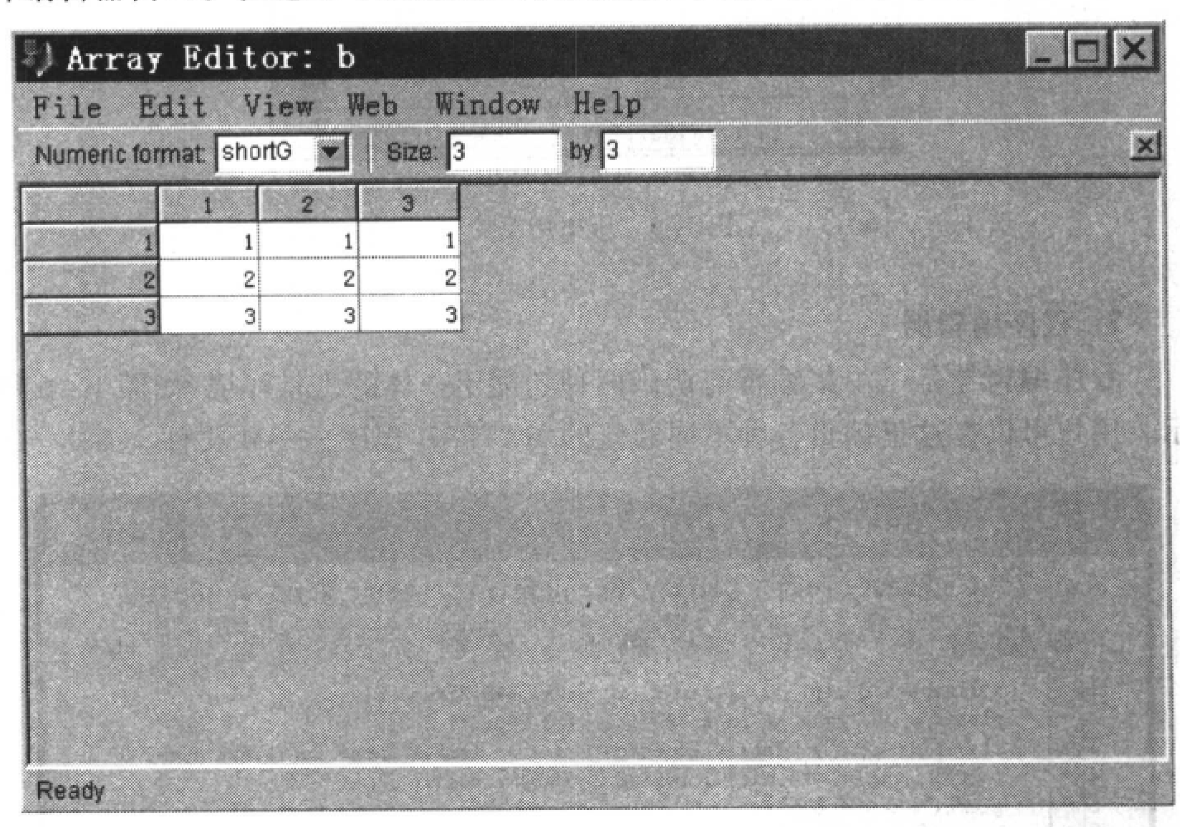


图 1-4 矩阵编辑器窗口

5. 当前工作路径下的目录窗口

在图 1-1 所示的窗口的左下部分是当前工作目录窗口(Current Directory)和历史指令窗口(Command History)。

当前工作目录窗口主要是保存在当前工作路径下的图形文件和指令文件，从图 1-1 中可以看到，当前的工作路径是 E:\matlab6.1\work\MATLAB。用户可以直接从这里选择要打开和要运行的文件。

6. 历史指令窗口

历史指令窗口主要是保存在指令窗口输入并运行过的指令、表达式等，如图

1－5 所示。这里显示的是以前在指令窗口曾输入的内容。必要时，用户可以直接把这些临时指令提取出来，放在指令窗口中使用。

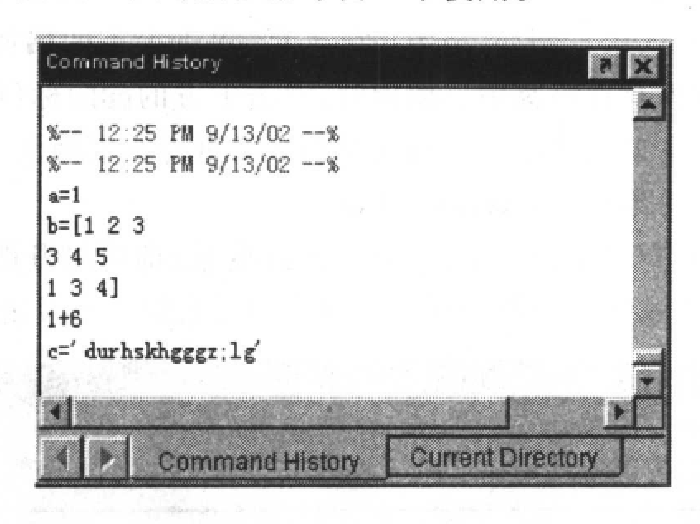


图 1－5　历史指令窗口

7. 程序编辑器

程序编辑器是一个集编辑与调试两种功能于一体的工具环境，如图 1－6 所示。用户可以在这里编辑各种不同功能的 MATLAB 程序——M 文件。

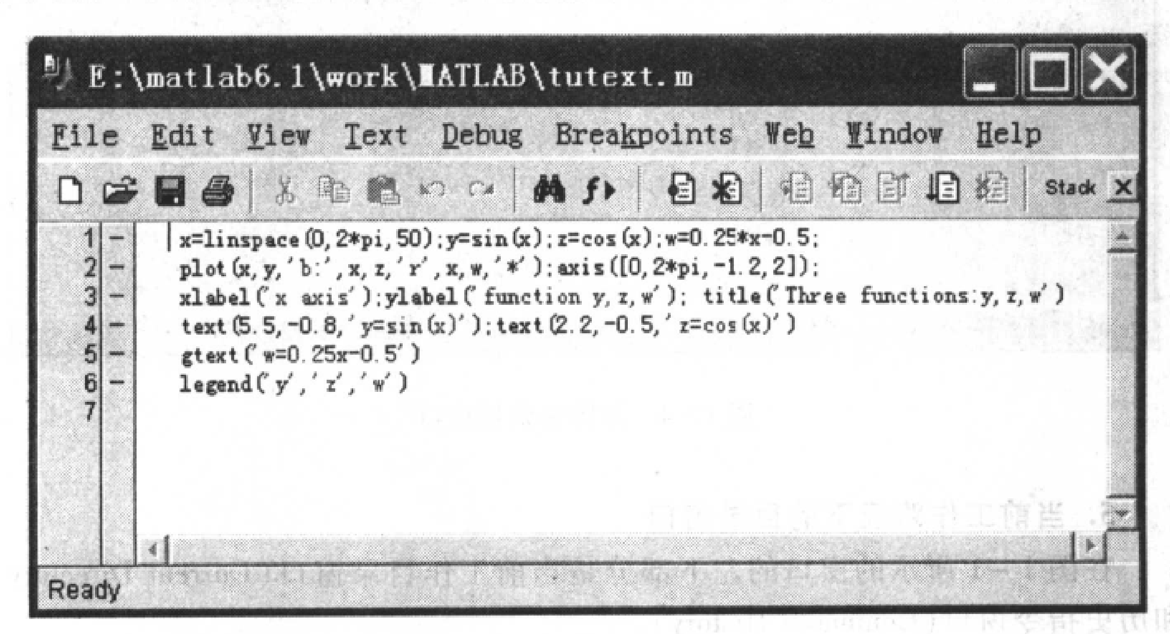


图 1－6　程序编辑器窗口

8. 帮助浏览器

在 View 菜单下选择 Help 项，就会出现帮助浏览器，左边是目录栏，右边是

帮助的内容。所有的帮助信息都可以在该浏览器中显示。而且用户可以对原有的帮助信息编辑取舍或加入自己的注解,形成自己的帮助文件。

MATLAB 6.x 把以上 8 项组件集成起来,构成桌面系统。而且各组件可以独立地构成视窗,具有自己的菜单和工具条,可以对视窗中的内容进行编辑和存储。

上面各组件视窗根据需要可以放在指令窗口内,也可以脱离指令窗口,还可以不显示,等需要时再打开。让用户使用起来更加方便、灵活。

在默认情形下,MATLAB 的工作台、工具箱、工作空间、历史窗口和当前工作目录都排列在指令窗口的左边,以方便用户使用。例如,查看当前的工作变量,显示工作路径等,不需要在指令窗口中输入指令,直接从工作空间和当前目录中就能看到。

在指令窗口中,如果用户要输入与前面相同的一些指令,在没有清除指令窗口变量的情形下,可以通过键盘的上下方向键进行选择,使输入的次数减少。如果已经清除指令窗口变量,用方向键就不行了。但是,现在还可以从历史指令窗口中去选择,找到指令后,拖动鼠标把指令复制到指令窗口,这样操作就很简单了。

此外,如果要打开一个保留在当前工作空间的 M 文件、图形窗口或图形,可以在当前工作目录窗口双击以打开指令窗口。

例如,要对某一个 M 文件进行查看或编辑,可以不在指令窗口中输入指令,也不必从菜单中通过几个选项来打开,只需在当前工作目录窗口直接双击即可。如现在打开一个已经存在的 M 文件 tutext.m,直接双击该文件,就弹出此文件编辑对话框,如图 1-6 所示。

1.3 MATLAB 管理指令

在指令窗口中,用户可以直接输入和执行指令。常用的窗口通用指令如表 1-1 所示。

表 1-1 MATLAB 工作窗口中的通用指令

指令	功能
clc	擦除指令窗口中显示的所有内容
clf	擦除当前图形窗口的图形
exit 或 quit	关闭并退出 MATLAB

续表

指　令	功　能
pack	整理内存碎片以扩大内存空间
cd	改变当前工作目录
dir	列出当前目录及该目录下的文件及子目录清单
clear	清除内存中的所有变量和函数
disp	在运行中显示变量或文字内容
echo	控制运行文字指令是否显示
hold	控制当前图形窗口对象是否被刷新
type	显示所指定文件的全部内容

在指令窗口中，为了便于对输入的指令和数据内容进行编辑，MATLAB 提供了控制光标位置和进行简单编辑的一些常用操作键和组合键，如表 1－2 所示。

表 1－2　通用操作键

键　名	功　能	键　名	功　能
↑	前寻式调出已输入过的指令行	Page Up	显示上一页
↓	后寻式调出已输入过的指令行	Page Down	显示下一页
←	光标左移一格	Delete	删除光标右侧字符
→	光标右移一格	Backspace	删除光标左侧字符
Home	光标移到当前行首	Esc	清除当前行的所有内容
End	光标移到当前行尾		

1.4　MATLAB 帮助系统

MATLAB 为用户提供了三种帮助功能，用户可以在帮助信息的指导下逐步熟练掌握 MATLAB 的使用方法。

1. 利用帮助菜单获取帮助信息

单击 MATLAB 工作窗口的菜单栏 Help 菜单项，弹出帮助菜单选项。

选择 Help Window 选项，可以打开 MATLAB 的主题窗口。该窗口列出了

MATLAB 的所有帮助主题,双击相关主题即可打开有关主题的进一步详细说明。

选择 Help Desk 选项,可以打开 MATLAB 帮助工作台。该工作台以超文本方式为用户提供帮助信息,从基本的入门帮助到工具箱的使用,用户只需单击工作台中的相关主题,即可获得该主题超文本格式的详细帮助信息。

2. 通过指令窗口获取帮助信息

用户也可以在指令窗口直接键入帮助指令来获得帮助。帮助指令如表 1-3 所示。

表 1-3 帮助指令

帮助指令	说明
help	列出 MATLAB 的所有帮助主题
helpwin	打开 MATLAB 的帮助主题窗口
helpdesk	打开 MATLAB 的帮助工作台
help help	打开有关使用帮助信息的帮助窗口
help 函数名(或主题名)	查询函数(或主题)的相关信息

3. 使用演示功能(Demo)

MATLAB 带有生动直观的演示程序,可以帮助用户形象直观地学习和理解 MATLAB 的使用方法和强大的功能。启动演示程序有下面几种方法。

(1) 在工作台和工具箱窗口(Launch Pad)中,列出了 MATLAB 和已经安装的各种工具箱。单击欲学习的工具箱前面的"+"号,在打开的功能项中,双击 Demos,即可打开演示程序。例如,要学习 MATLAB 的基本功能,可以单击 Launch Pad 窗口内 MATLAB 项前的"+"号,打开的功能列表如图 1-7 左上方所示,双击其中的 Demos,即可打开图 1-7 右边的演示窗口。在演示窗口中,选中左面的演示主题,如 Graphics,在右下方会出现此主题下的具体项目列表,选择其中的一个项目,如 2-D Plots 后,单击下方的运行 Run 2-D Plots 按钮,即可学习二维图形的绘制方法。

(2) 选择 Help 菜单的 Demos 选项,可打开如图 1-7 右边所示的 MATLAB 演示窗口。

(3) 在指令窗口中键入指令 demo,同样可以打开 MATLAB 演示窗口。

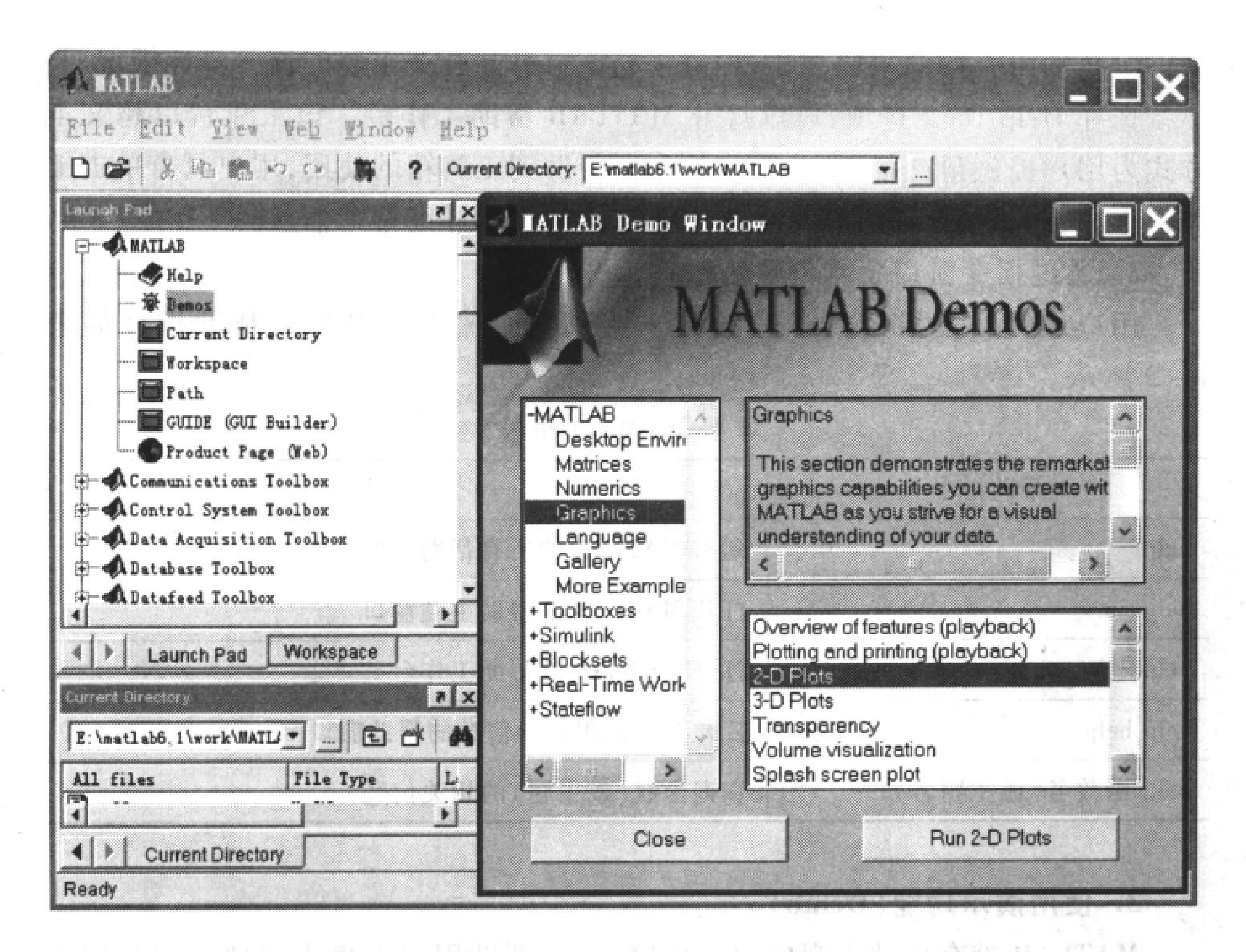


图 1－7　在 Launch Pad 窗口中打开 MATLAB 演示程序

第二章 MATLAB 的数值计算

作为学习 MATLAB 的基础，本章介绍最基本的数值计算功能。主要包括 MATLAB 的变量与表达式，矩阵和向量的创建、保存与运算，MATLAB 常用矩阵和向量的数学运算函数的使用，关系运算与逻辑运算的规则，多项式及其运算。

2.1 MATLAB 的变量与表达式

1. 变量的定义

和其他高级语言一样，MATLAB 也是使用变量来保存信息的，变量由变量名表示。变量的命名是以字母开头，后接字母、数字或下划线的字符串，最多 31 个字符，且区分大小写。如 asum、Asum 和 ASUM 表示的是三个不同的变量。

MATLAB 中的变量一般无需事先定义。一个程序中的变量以其名称在操作语句中第一次合法出现而定义。如果这个变量已经存在，那么 MATLAB 将改变它的内容，如 a =2.5 定义了一个变量 a，并给它赋值 2.5；如果再输入 a =3，那么变量 a 的值就变为 3。

2. 变量的类型

MATLAB 提供了 6 种基本的变量类型，它们是：

(1) 双精度型(doube)。　(2) 字符型(char)。

(3) 稀疏型(sparse)。　(4) 单元型(cell)。

(5) 结构型(struct)。　(6) 8 位型(uint8)。

在这些类型中，用户最常使用的是双精度型和字符型。

MATLAB 中不需要专门定义变量的类型，系统可以自动根据表达式的值或输入的值来确定变量的数据类型。但是，如果使用和原来定义的变量一样的名字赋值，原变量的内容将自动被覆盖，系统不会给出出错信息。因此，在使用变量时要自觉地避免重复。

3. 固定的内部变量

在 MATLAB 中有一些系统默认的固定内部变量，如表 2－1 所示，即在

MATLAB 语句中若出现表中的变量名，则系统就将赋予其默认值。

表 2-1 MATLAB 的固定内部变量

变量名	默认值	变量名	默认值
i 或 j	复数虚单位 $\sqrt{-1}$	inf 或 Inf	无穷大∞（1/0）
pi	圆周率 π	NAN	非数（0/0、0 * inf、inf/inf）
ans	缺省变量名。以操作中的最近应答作为它的值	realmax	MATLAB 的最大浮点数 10^{+308}
eps	计算浮点数的误差限 2^{-52}（10^{-16}）	realmin	MATLAB 的最小浮点数 10^{-308}
nargin	函数输入变元的个数	nargout	函数输出变元的个数

4. 数值的输出显示

在 MATLAB 内部，每一个数值元素都是用双精度来表示和存储的，有 16 位数字，其数值有效范围约为 $10^{-308} \sim 10^{+308}$。

在进行数值输入和输出时，用户可以改变 MATLAB 在屏幕上显示的格式。如果参加运算的每一个元素均为整数，则 MATLAB 将用不加小数点的纯整数格式显示运算结果，否则，按设定的输出格式显示结果。建议读者采用 short 或 short g 输出格式。数值的输出格式见表 2-2。

表 2-2 数值输出显示的 10 种格式

数据输出格式	显示形式	说　明
short（默认）	30.7143 3.0714e+003 -40000	6 位十进制定点表示（定 4 位小数，对小于 10^9 的整数不显示小数，对大于 100 的小数和大于 10^9 的整数自动按 short e 格式显示）
long	30.71428571428572 3.071428571428572e+003 -40000	16 位十进制定点表示（对小于 10^9 的整数不显示小数，对大于 100 的小数和大于 10^9 的整数自动按 short e 格式显示）
short e	3.0714e+001 3.0714e+003 -40000	5 位十进制浮点指数表示（对小于 10^9 的整数只显示整数）
long e	3.071428571428572e+001 3.071428571428572e+003 -40000	16 位十进制浮点指数表示（对小于 10^9 的整数只显示整数）

续表

数据输出格式	显示形式	说　明
short g	30.714 3071.4 -40000	5 位十进制紧凑表示(小数位随整数位数变化,且自动取最少小数位,对小于 10^9 的整数不显示小数,对大于 10^5 的小数和大于 10^9 的整数自动按 short e 格式显示)
long g	30.7142857142857 3071.42857142857 -40000	15 位十进制紧凑表示(小数位随整数位数变化,且自动取最少小数位,对小于 10^9 的整数不显示小数,对大于 10^{15} 的小数和大于 10^9 的整数自动按 long e 格式显示)
hex	403eb6db6db6db6e 40a7fedb6db6db6e c0e3880000000000	16 位十六进制表示
bank	30.71 3071.43 -40000.00	用元、分(美制)定点表示(定点 2 位小数)
+	+ -	用 +、- 和空格表示正数、负数和零
rational	215/7 21500/7 -40000	近似的分数表示

输出显示格式的改变方法:

(1) 选 MATLAB 指令窗口顶部的菜单项 file\preferences,打开图 2-1 所示的 preferences 对话框,在左边 Command Window 被选中的情况下,对右边的 Numeric format 栏进行选择。下方 Numeric display 栏是显示的不同形式。其中 loose 为稀疏形式(系统默认形式);compact 为紧凑形式。

(2) 在指令窗口中输入 format 指令来指定数值的输出格式。例如,format short 表示其后的数据以 short 格式输出显示,直到出现下一个 format 指令。

5. 矩阵与向量

矩阵是 MATLAB 进行数据处理和运算的基本单元,它由 $n\times m$ 个元素组成,MATLAB 的大部分运算或指令都是在矩阵运算的意义下执行的。一个变量就是一个矩阵,若某变量是 1×1 的矩阵(只有一个元素),它就是常说的标量;若某变量是 $n\times 1$ 或 $1\times m$ 的矩阵(只有一列或一行),它就是常说的向量。

从运算的角度看,MATLAB 的矩阵运算和向量运算有显著不同。矩阵运算

图 2－1　preferences 对话框

是从矩阵的整体出发,依照线性代数的运算规则进行。而向量运算是从向量中的元素出发,针对每个元素进行计算的。

6. 字符串与字符串变量

在 MATLAB 中,字符串是用单引号括起来的字符序列,例如:

```
>>xx='character string'
```

输出结果为

```
xx =
        character string
```

其中,>>是 MATLAB 指令窗口的指令行提示符。

MATLAB 将字符串当作一个行向量,每个元素对应一个字符,其值为字符的 ASCII 码值。于是,上面建立的字符串变量 xx 是 1×16 的矩阵。

对于字符串的写法还应注意:

若字符串中的字符含有单引号，则该单引号字符要用两个单引号来表示，例如：

```
>>disp('I''m a teacher')
% disp 的作用是将字符串变量的字符直接显示出来
```

输出为

```
I'm a teacher
```

对较长的字符串可以用字符串数组表示，即用[ ]括起来，例如：

```
>>s='Hello';ss=[s,' MATLAB']
% 分号在语句结尾的作用是将计算结果存入内存但不显示
```

输出为

```
ss =
    Hello MATLAB
```

7. 表达式

MATLAB 采用的是表达式语言，用户输入的语句由 MATLAB 系统解释运行。MATLAB 的语句是由表达式和变量组成的，有如下两种最常见的形式：

- 表达式
- 变量 = 表达式

表达式由变量、数字、函数和运算符组成。几乎所有的运算都必须借助于表达式进行。

在第一种形式中，表达式运算后产生的结果，系统会自动赋给名为 ans 的变量。ans 是一个默认的变量名，它会在以后的类似操作中被自动覆盖掉。所以，对于后续计算有用的结果一定要记录下来，这就要使用第二种形式。

在第二种形式中，系统会将表达式计算后产生的结果赋给等号左边的变量，并放入内存中。

8. 函数

MATLAB 为用户提供了丰富且功能各异的内部函数，用户可以直接调用这些内部函数来进行数据处理。函数由函数名和变元组成，函数调用的格式为

函数名(变元)

注意：函数名一律由小写字母组成。在后面的章节中会大量用到这些函数。例如，a = sin(b)表示计算 b 的正弦函数值并将其赋值给变量 a。

用户也可以创建自己的函数文件，其详细内容在第三章介绍。

9. 运算符

MATLAB 的基本运算为算术运算、关系运算、逻辑运算和特殊运算等。每一

类运算都有自己专用的运算符。表 2－3 列出了这些运算符及其功能。它们的详细使用方法请看 2.3 节。

表 2－3　各种运算符

运算符		意　义	运算符		意　义
算术运算符	+	加(A＋B)。若 A、B 为相同维数的矩阵,二者对应元素相加;若一个为标量,矩阵元素均加此标量	关系运算符	＝＝	相等
	－	减(A－B)。若 A、B 为相同维数的矩阵,二者对应元素相减;若一个为标量,矩阵元素均减此标量		～＝	不等于
	*	矩阵乘(A＊B)。两矩阵的维数应满足乘法的定义		>	大于
	.*	向量乘(A.＊B)。相同维数的两个向量的对应元素相乘		<	小于
	^	矩阵乘方(A^B)。A、B 至少有一个是标量		>＝	大于等于
	.^	向量乘方(A.^B)。同维向量对应元素乘方		<＝	小于等于
	\	矩阵左除(A\B)。方程 A＊X＝B 的解 X	特殊运算符	:	冒号(产生数组)
	.\	向量左除(A.\B)。同维向量对应元素相除		[ ]	方括号(生成矩阵或数组)
	/	矩阵右除(A/B)。方程 X＊A＝B 的解 X		;	分隔矩阵行,若在语句末则表示不显示结果
	./	向量右除(A./B)。同维向量对应元素相除		%	注释符
	'	共轭转置(A')。矩阵的共轭转置		...	续行符
	.'	转置(A.')。矩阵与向量的转置		,或空格	分隔矩阵列
逻辑符	&	逻辑与		( )	圆括号(函数调用)
	|	逻辑或		{ }	花括号(生成 cell)
	～	逻辑非		＝	赋值符

例如,计算表达式 $\frac{2\sin 85^\circ}{1+\sqrt{5}+3i}$ 的值,将结果赋给变量 A,并显示计算结果。在 MATLAB 指令窗口输入

```
>>A=2*sin(85*pi/180)/(1+sqrt(5)+3*i)
```

输出结果为

```
A =   0.3311 - 0.3070i
```
[①]

2.2 矩阵的创建与保存

在 MATLAB 中,矩阵是运算的基本单元,而且按复数定义。MATLAB 中所有矩阵事先都不必定义维数大小,系统会根据用户的输入自动配置,在运算中自动调整矩阵的维数。

在 MATLAB 中,矩阵可通过下列三种方法之一创建:直接输入法,利用 MATLAB 的内部函数创建,利用矩阵编辑器创建和修改矩阵。

1. 直接输入法创建矩阵

(1) 实数矩阵。

对于维数较小的实数矩阵,创建它最简单的方法是从键盘直接输入。具体方法为:将矩阵的元素用方括号"[ ]"括起来,按矩阵行的顺序输入各元素,同一行的各元素之间用空格" "或逗号","分隔,行与行之间用分号";"或换行符隔开。

例如,在指令窗口内直接输入指令:

```
>>a=[1,2,3;4,5,6;7,8,9]
```

按 Enter 键执行后,将创建的 3×3 矩阵赋值给变量 a,并在屏幕上显示

```
a =
     1     2     3
     4     5     6
     7     8     9
```

(2) 复数矩阵。

上面矩阵 a 的元素是实数,它实际上是虚部为零的复数。MATLAB 对复数的操作非常方便。例如,6+5i 与 6+5j 表示的是同一个复数,注意不可以写作 6+i5或 6+j5,为了避免出错,也可以写成 6+i*5 或 6+j*5。有两种方法建立

---

① 实际显示形式为

```
A =
    0.3311 -0.3070i
```

为节省纸张,以下将省去显示结果中的一些换行符输出。

复数矩阵。

一种方法是将其元素逐个赋予复数，例如：

```
>>a=exp(2);  % 分号的作用是不显示计算结果
>>b=[1,2+i*a,a*sqrt(a);sin(pi/4),a/5,3.5+6i]
```

输出结果为

```
b =
    1.0000          2.0000 + 7.3891i       20.0855
    0.7071          1.4778                  3.5000 + 6.0000i
```

另一种方法是将复数的实部和虚部矩阵分别赋值，例如：

```
>>R=[1,2,3;4,5,6];I=[6,7,8;9,10,11];
>>ri=R+i*I
```

输出结果为

```
ri =
    1.0000 + 6.0000i    2.0000 + 7.0000i    3.0000 + 8.0000i
    4.0000 + 9.0000i    5.0000 +10.0000i    6.0000 +11.0000i
```

2. 利用 MATLAB 内部函数建立数值矩阵

MATLAB 提供了许多函数用于创建一些常用的特殊矩阵，表 2-4 是常用的创建矩阵函数。

表 2-4 MATLAB 创建矩阵函数

函数调用格式	说　明
zeros(m,n)	生成一个元素全部为 0 的 m 行 n 列矩阵
ones(m,n)	生成一个元素全部为 1 的 m 行 n 列矩阵
eye(m,n)	生成一个主对角线元素为 1、其他元素为 0 的 m 行 n 列矩阵
rand(m,n)	生成一个元素在 0 和 1 之间均匀分布的 m 行 n 列随机矩阵
randn(m,n)	生成一个元素为 0 均值单位方差正态分布的 m 行 n 列随机矩阵
pascal(n)	生成一个 n 维帕斯卡矩阵
magic(n)	生成一个 n 维魔方矩阵

3. 用工作空间的矩阵编辑器创建和修改矩阵

当输入的矩阵很大，不适合用手工直接输入时，可以利用工作空间下的矩阵编辑器(Matrix Editor)来创建和修改矩阵。在调用矩阵编辑器之前，需要预先定义一个变量，无论是数值还是矩阵均可。例如：

```
>>a=1;      % 定义一个名为 a 的变量
```

在工作空间(Workspace)窗口内,就可以看到这个变量 a,如图 2-2 所示。双击变量 a,即可打开如图 2-3 所示的矩阵编辑器窗口。这个窗口现在显示的是刚才定义的 1 行 1 列的矩阵。改变 size ~ by(矩阵的行和列)文本框内的数值并按 Enter 键,即可把它裁剪或扩展为任意规模的矩阵,且系统会自动地把扩展部分的元素设置为 0。例如,图 2-4 是把原矩阵扩展为 10 行 8 列的新矩阵。要想对这个矩阵中的元素进行修改,只需选中此元素,再输入新的数值或表达式即可。

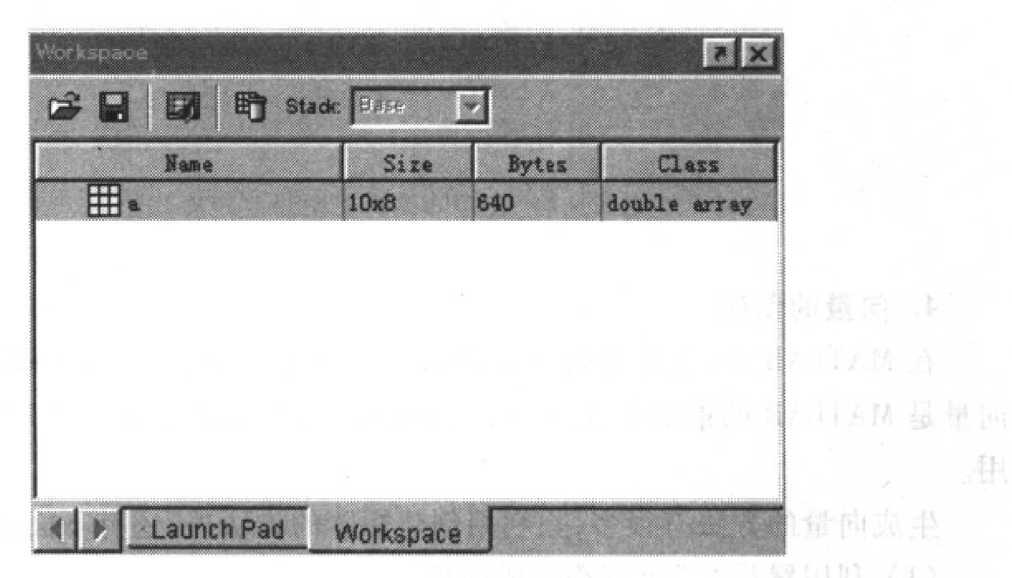


图 2-2 工作空间窗口

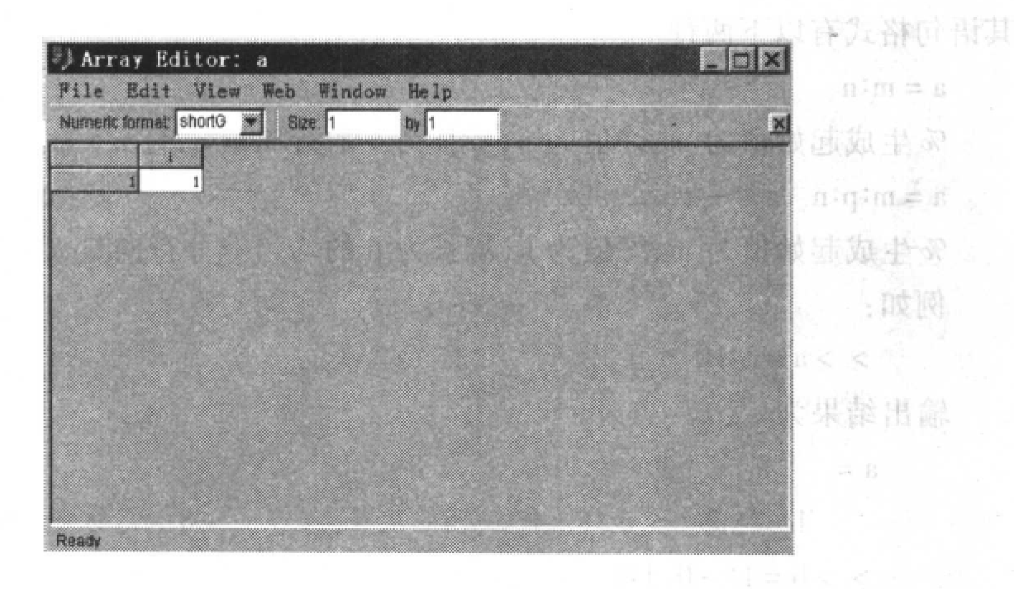


图 2-3 矩阵编辑器窗口

Array Editor: a

File Edit View Web Window Help

Numeric format: shortG Size: 10 by 8

	1	2	3	4	5	6	7	8
1	1	0	4	5	0	0	0	7
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

Ready

图 2-4 在矩阵编辑器中修改矩阵

4. 向量的创建

在 MATLAB 中,仅有 1 行或 1 列的矩阵被称为向量,它是矩阵的一种特例。向量是 MATLAB 的重要概念之一,它在信号的表示和处理方面起着重要的作用。

生成向量的方法有很多,除利用创建矩阵的方法外,还有下面几种方法。

(1) 利用冒号":"运算符生成向量。

冒号运算用于生成等步长(均匀等分)的行向量,在已知步长情况下采用。其语句格式有以下两种。

```
a = m:n
% 生成起始值为 m,终值为 n,步长为 1 的行向量 a,且 n > m
a = m:p:n
% 生成起始值为 m,终值为 n,步长为 p 的均匀等分行向量 a
```

例如:

```
>> a = 1:10
```

输出结果为

```
a =
     1     2     3     4     5     6     7     8     9    10
```

```
>> b = 1: -0.1:0
```

输出结果为

```
b =
```

```
1  0.9  0.8  0.7  0.6  0.5  0.4  0.3  0.2  0.1  0
```

（2）利用向量生成函数来创建向量。

在已知数据总数的情况下，常使用线性间隔向量函数 linspace 和对数间隔向量函数 logspace 生成向量。linspace 函数的指令格式为

```
a = linspace(m,n)
% 生成把 m 到 n 间的数 100 等分的行向量，并赋值给变量 a
a = linspace(m,n,s)
% 生成把 m 到 n 间的数 s 等分的行向量，并赋值给变量 a
```

例如：

```
>>x = linspace(1,3,6)
```

输出结果为

```
x =
    1      1.4      1.8      2.2      2.6      3
```

logspace 函数的指令格式：

```
a = logspace(m,n)
% 生成从 10^m 到 10^n 间的 50 个对数间隔点的行向量
a = logspace(m,n,s)
% 生成从 10^m 到 10^n 间的 s 个对数间隔点的行向量
```

例如：

```
>>a = logspace(2,5,15)
```

输出结果为

```
a =
  Columns 1 through 8
    100   163.79   268.27   439.4   719.69   1178.8   1930.7   3162.3
  Columns 9 through 15
  5179.5   8483.4   13895   22758   37276   61054   1e+005
```

5. 矩阵的保存和提取

对于大型计算，经常有大量数据的矩阵需要保存以便在下次使用时提取。MATLAB 提供了对工作空间中存在的矩阵进行保存和提取的多种方法。

使用指令 save，调用格式为

save　文件名　矩阵名

例如：

```
save myfile a b
```

% 把矩阵 a 和 b 以文件名 myfile.mat 保存到默认的文件夹中

使用 save 指令,可以将指定的矩阵保存到以 mat 为后缀文件名的文件中,并自动放到默认的 work 文件夹中。如果要保存整个工作空间中的矩阵,可以直接给出指令

save 文件名

将保存的矩阵取回到工作空间的指令是 load,调用格式为

load 文件名

例如:

load myfile

% 把 work 文件夹中的文件 myfile.mat 加载到工作空间

矩阵的保存和提取也可使用 1.2 节中介绍的方法,直接在工作空间窗口进行编辑操作。也可以通过主菜单 File\Save Workspace As,打开 Save Workspace Variables 对话框来完成数据的存储,用 File\Open 来完成数据的加载。

2.3 MATLAB 常用的矩阵运算函数

MATLAB 用运算符 * 、/、\和^表示矩阵的乘除和求幂运算,用函数 expm、logm 和 sqrtm 表示矩阵的指数、自然对数和开方运算。如果一个标量(1×1 矩阵)与另一矩阵相加、减、乘、除,则表示该标量与矩阵的每一个元素的相应运算。如果只是矩阵对应元素的运算,为了与矩阵运算相区别,要在运算符 * 、/、\和^前加符号“.”,可称为点运算。常用的运算符参看表 2-3。下面是一些示例。

```
>> a=[1,2,3;2,3,4;4,5,6];b=[0,3,9;4,8,2;2,3,7];
% 定义矩阵 a,b
>> c=a+b     % 计算两矩阵 a 和 b 的和
c =
     1     5    12
     6    11     6
     6     8    13
>> d=a*b     % 计算两矩阵 a 和 b 的乘积
d =
    14    28    34
    20    42    52
    32    70    88
```

```
>> e = a/c
% 计算矩阵 a 和 c 的右除,等于 a * inv(c),其中 inv(c)为 c 的逆矩阵
e =
     0.1299    0.0634    0.0816
     0.0242    0.0816    0.2477
    -0.1873    0.1178    0.5801
>> f = c\a    % 计算矩阵 c 和 a 的左除,等于 inv(c) * a
f =
     0.7583     0.6858     0.6133
    -0.3142    -0.2085    -0.1027
     0.1511     0.1964     0.2417
>> g = a'     % 计算矩阵 a 的共轭转置,对于实矩阵,也是转置运算
g =
     1     2     4
     2     3     5
     3     4     6
>> a1 = a^2   % 计算矩阵 a 的平方
a1 =
    17    23    29
    24    33    42
    38    53    68
>> x = [1,2,3,4];y = [3,8,4,1];x1 = x. * y
% 定义向量 x 和 y,并计算向量的点乘
x1 =
     3    16    12     4
>> x2 = x. \y      % y(i)/x(i)
x2 =
    3.0000    4.0000    1.3333    0.2500
>> x3 = x./y       % x(i)/y(i)
x3 =
    0.3333    0.2500    0.7500    4.0000
```

除了用运算符进行简单的运算以外,MATLAB 提供了许多矩阵和向量的运算函数来完成功能各异的运算任务,表 2-5 ~ 表 2-7 列出了一些运算函数,使

用时要注意它们的意义和运算规则。

表 2-5 常用矩阵运算函数

函数名称	功能简介
norm(A,1)	矩阵 A 的 1-范数
norm(A)	矩阵 A 的 2-范数
norm(A,inf)	矩阵 A 的无穷大-范数
rank(A)	矩阵 A 的秩
trace(A)	矩阵 A 的迹
sum(A,k)	矩阵 A 的列求和或行求和。k=1 按列求和(默认),产生一行向量;k=2 按行求和,产生一列向量
prod(A,k)	矩阵 A 的列乘积或行乘积。k=1 按列求乘积(默认),产生一行向量;k=2 按行求乘积,产生一列向量
max(A,[],k)	k=1 求矩阵 A 各列元素的最大值,得一行向量;k=2 求各行元素的最大值,得一列向量
min(A,[],k)	k=1 求矩阵 A 各列元素的最小值,得一行向量;k=2 求各行元素的最小值,得一列向量
mean(A,[],k)	k=1 求矩阵 A 各列元素的平均值,得一行向量;k=2 求各行元素的平均值,得一列向量
median(A,[],k)	k=1 求矩阵 A 各列元素的中值,得一行向量;k=2 求各行元素的中值,得一列向量
std(A,1,k)	矩阵 A 的方差。k=1 按列计算(默认),产生一行向量;k=2 按行计算,产生一列向量
sort(A,k)	矩阵 A 升序排序。k=1 按列计算(默认),得一行向量;k=2 按行计算,得一列向量
expm(A)	矩阵 A 的指数运算 e^A
logm(A)	矩阵 A 的自然对数
sqrtm(A)	矩阵 A 的平方根

表 2-6 矩阵结构形式提取和变换函数

函数名称	功能简介
diag(A)	矩阵 A 的对角阵,产生一行向量
eig(A)	方阵 A 的特征值和特征向量
inv(A)	满秩方阵 A 的逆矩阵
compan(A)	矩阵 A 的伴随阵
rot90(A)	矩阵 A 整体逆时针旋转 90°
rot90(A,k)	矩阵 A 整体逆时针旋转 k×90°
fliplr(A)	矩阵 A 左右翻转
flipud(A)	矩阵 A 上下翻转
tril(A,k)	矩阵 A 第 k 条对角线及以下的元素保留,其余为 0
tril(A)	矩阵 A 主对角线及以下的元素保留,其余为 0(默认主对角线 k=1)
triu(A,k)	矩阵 A 第 k 条对角线及以上的元素保留,其余为 0
triu(A)	矩阵 A 主对角线及以上的元素保留,其余为 0(默认主对角线 k=1)
size(A)	计算矩阵 A 的行数和列数,结果为 1×2 的向量
length(A)	计算矩阵 A 的行数与列数中最大者,结果为一个标量
det(A)	方阵 A 的行列式,结果为一个标量

表 2-7 向量运算基本函数

类别	函数	说明	函数	说明
三角函数	sin	正弦	cos	余弦
	tan	正切	cot	余切
	asin	反正弦	acos	反余弦
	atan	反正切	atan2(x,y)	四象限反正切
	acot	反余切	sinh	双曲正弦
	cosh	双曲余弦	tanh	双曲正切
	coth	双曲余切	asinh	反双曲正弦
	acosh	反双曲余弦	atanh	反双曲正切
	acoth	反双曲余切	sec	正割
	csc	余割	asec	反正割
	acsc	反余割	sech	双曲正割
	csch	双曲余割	asech	反双曲正割
	acsch	反双曲余割		
其他函数	exp	以 e 为底的指数	log	自然对数
	log2	以 2 为底的对数	log10	以 10 为底的对数
	pow2	2 的幂	sqrt	平方根
	sign	符号函数		
	abs	绝对值和复数的模	angle	复数的辐角
	real	复数的实部	imag	复数的虚部
	conj	复数的共轭	rat(x)	将实数 x 化为分数表示
	round	四舍五入为整数	fix	向 0 方向整数
	floor	舍去正小数至最近整数	ceil	加入正小数至最近整数
	rem(x,y)	求 x 除以 y 的余数	mod(x,y)	求 x 除以 y 的正余数
	lcm(x,y)	整数 x 和 y 的最小公倍数	gcd(x,y)	整数 x 和 y 的最大公因数

下面给出一些示例。

```
>> aa=[1,0,0,0;0,2.5,0,3;0,0,4,0;0,0,0,1];   % 定义方阵 aa
>> aa1=det(aa)      % 计算矩阵 aa 的矩阵行列式
```

```
aa1  =   10
>> aa2 = trace(aa)   % 计算矩阵 aa 的迹
aa2  =   8.5000
>> aa3 = rank(aa)   % 计算矩阵 aa 的秩
aa3  =   4
>> aa4 = eig(aa)   % 计算矩阵 aa 的特征值向量(结果是一个列向量)
aa4  =
    1.0000
    2.5000
    4.0000
    1.0000
>> aa5 = inv(aa)   % 计算满秩矩阵 aa 的逆矩阵
aa5  =
    1.0000         0         0         0
         0    0.4000         0   -1.2000
         0         0    0.2500         0
         0         0         0    1.0000
>> x =[1,2,3,4];x1 = sin(x) % 计算向量 x 每个元素的正弦函数值
x1  =
    0.8415    0.9093    0.1411   -0.7568
>> x2 = sqrt(x) % 计算向量 x 每个元素的平方根
x2  =
    1.0000    1.4142    1.7321    2.0000
```

2.4 关系运算与逻辑运算

在 MATLAB 中,所有关系运算及逻辑运算都是按向量运算规则定义的。

1. 关系运算

关系运算符参看表 2-3,关系运算的规则是:

(1) 参与关系运算的矩阵必须是两个相同维数的矩阵或其中之一是标量。

(2) 当参与运算的矩阵是两个相同维数的矩阵 A 和 B 时,关系运算的结果是将矩阵 A 和 B 下标相同的对应元素逐一进行关系比较,若关系成立则比较结果值为 1,若关系不成立则比较结果值为 0。也即关系运算的结果是生成一个与

A 和 B 维数相同的矩阵,其元素值为 0 或 1。

(3) 当参与运算的矩阵之一为标量时,关系运算的结果是将矩阵的每一个元素与该标量逐一进行关系比较,若关系成立则比较结果值为 1,若关系不成立则比较结果值为 0。

(4) 关系运算比算术运算具有更高的优先权。

```
>>a=[1,2,3;2,1,3;3,2,1];b=[2,2,2;2,2,2;2,2,2];
% 输入矩阵 a 和 b
>> c=a>=b
% 比较 a 中元素是否大于等于 b 中对应元素,结果放入 c 中
```

输出结果为

```
c =
    0    1    1
    1    0    1
    1    1    0
>> d=a==2
% 比较 a 中的元素是否等于 2,结果放入 d 中
```

输出结果为

```
d =
    0    1    0
    1    0    0
    0    1    0
```

2. 逻辑运算

逻辑运算符请看表 2-3。在逻辑运算中,所有非零元素的逻辑值为真,用代码 1 表示,值为零的元素的逻辑值为假,用代码 0 表示。逻辑运算的规则是:

(1) 参与逻辑运算的矩阵必须是两个相同维数的矩阵或其中之一是标量。

(2) 当参与运算的矩阵是两个相同维数的矩阵 A 和 B 时,逻辑运算的结果是将矩阵 A 和 B 下标相同的对应元素只要进行逻辑运算,若逻辑运算的值为真,则运算结果值为 1;若逻辑运算的值为假,则运算结果值为 0。即逻辑运算的结果是生成一个与 A 和 B 维数相同的矩阵,其元素值为 1 和 0。

(3) 当参与运算的矩阵之一是标量时,逻辑运算的结果是将矩阵的每一个元素与该标量进行逻辑运算,若逻辑运算值为真,则运算结果值为 1;若逻辑运算的值为假,则运算结果值为 0。

三种逻辑运算的真值表如表 2-8 所示。

表 2-8 逻辑运算真值表

<table>
<tr><th>a</th><th>b</th><th>a&b(与)</th><th>a|b(或)</th><th>~a(非)</th></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
</table>

下面是逻辑运算的示例。

```
>> a=[0,1,0,2;0,50,1,2;0,6,0,7];b=[0,1,0,0;0,5,0,0;1,0,5,0];
>> c=a&b  % a中元素和b中对应元素进行与运算,结果放在c中
c =
     0     1     0     0
     0     1     0     0
     0     0     0     0
>> d=a|b  % a中元素和b中对应元素进行或运算,结果放在d中
d =
     0     1     0     1
     0     1     1     1
     1     1     1     1
>> e=~a  % a中元素进行非运算,结果放在e中
e =
     1     0     1     0
     1     0     0     0
     1     0     1     0
```

3. 关系与逻辑函数

除了上面的关系与逻辑操作符外,MATLAB 还提供了大量的其他关系与逻辑函数,参见表 2-9。

表 2-9 关系与逻辑测试函数

函数名称	功能简介
xor(x,y)	异或运算。x 和 y 都是零(假)或都是非零(真)返回零,否则返回 1
any(x)	向量 x 中任何元素非零返回 1;矩阵 x 中每一列有非零元素返回 1
all(x)	向量 x 中任何元素非零返回 1;矩阵 x 中每一列任何元素非零返回 1
find	找出向量或矩阵中非零元素的位置和标识
finite	元素值大小有限返回 1
isempty	参量为空返回 1
isglobal	参量为全局变量返回 1
isinf	元素无穷大返回 1
isnan	元素为不定值返回 1
isreal	参量无虚部返回 1
isspace	参量为空格返回 1
isstr	变量为字符串返回 1
exist	检查变量在工作空间是否存在,若存在返回 1

2.5 多项式及其运算

MATLAB 使用行向量来表示多项式的系数,行向量中各元素按照多项式项的次数从高到低排列。对于多项式

$$p(x) = a_0x^n + a_1x^{n-1} + a_2x^{n-2} + \cdots + a_{n-1}x + a_n$$

可以用行向量 $p = [a_0, a_1, a_2, \cdots, a_{n-1}, a_n]$ 来表示。

例如,多项式 $p(x) = x^3 + 5x^2 - 6$,可以用 $p = [1,5,0,-6]$ 来表示,因而可以在指令窗口中直接输入。注意:对于系数为 0 的项,必须用 0 填充。

MATALAB 提供了一些处理多项式的基本函数,如求多项式的值、根和微分等。除此之外,MATLAB 也提供了一些较高级的函数用于处理多项式,如拟合曲线和部分分式展开等。

表 2-10 列出了较常用的多项式函数。

表 2-10 MATLAB 中多项式函数

函数名称	说明
conv(A,B)	求多项式 A 和 B 的乘法,两序列的卷积
[q,r] = deconv(A,B)	求矩阵多项式除法,A 为被除式,B 为除式,q 为商,r 为余数
roots(A)	求多项式的根,结果为列向量
poly(r)	由多项式的实根列向量 r 求多项式系数
poly(A)	计算矩阵 A 的特征多项式系数
polyval(A,x)	求多项式 A 当未知数为 x 时的值
polyder(A)	求多项式 A 的一阶导数
[r,p,k] = residue(B,A)	部分分式展开式,当分母多项式的根互不相等时,有 $\frac{B(x)}{A(x)}=\frac{r_1}{x-p_1}+\frac{r_2}{x-p_2}+\cdots+\frac{r_n}{x-p_n}+k$
[B,A] = residue(r,p,k)	对多项式的部分分式作逆运算

下面是一些示例。

```
>> a=[2,4,3];        % 输入多项式 a=2x^2+4x+3
>> b=[-1,5,0,4];     % 输入多项式 b=-x^3+5x^2+4
>> c=conv(a,b)       % 计算多项式 a 和 b 的乘积
c =
    -2    6    17    23    16    12
```

即

$$c(x) = -2x^5+6x^4+17x^3+23x^2+16x+12$$

```
>> [q,r] =deconv(c,b)     % 计算多项式 c 和 b 的商
q =
     2     4     3
r =
     0     0     0     0     0     0
```

商多项式 $q(x)=2x^2+4x+3$,它与 $a(x)$ 相同;余多项式 $r=0$ 表示整除。

```
>> p=[1,5,0,-6];      % 输入多项式 p=x^3+5x^2-6
>> r=roots(p)         % 计算多项式 p 的根
```

```
r =
    -4.7321
    -1.2679
     1.0000
>> p1 =poly(r) % 求根列向量组成的多项式的系数
p1 =
    1.0000    5.0000    -0.0000    -6.0000
>> polyval(p,2)  % 计算 x=2 时多项式 p 的值
ans =
    22
>> p2 =polyder(p) % 计算多项式 p 的导数
p2 =
     3     10     0
```

结果为 $p_2(x)=3x^2+10x$

```
>> A=[1,3,6;3,-2,0;9,1,1];  % 输入方阵 A
>> p=poly(A)  % 计算矩阵 A 的特征多项式
p =
    1.0000   -0.0000   -66.0000   -115.0000
```

结果为 $p(x)=x^3-66x-115$

```
>> r=roots(p) % 计算特征多项式 p 的根
r =
     8.8850
    -7.0488
    -1.8362
>> q=eig(A)     % 用函数 eig 直接求矩阵 A 的特征值
q =
     8.8850
    -7.0488
    -1.8362
```

两种方法求得的特征值相同。

```
>> B=[1,3,4,0,5];     % 输入分子多项式 B=x^4+3x^3+4x^2+5
>> A=[1,-5,6];      % 输入分母多项式 A=x^2-5x+6
>> [r,p,k]=residue(B,A)      % 对有理分式展开为部分分式
```

```
r =
    203.0000
   -61.0000
p =
   3.0000
   2.0000
k =
   1     8     38
```

即

$$\frac{x^4+3x^3+4x^2+5}{x^2-5x+6}=\frac{203}{x-3}+\frac{-61}{x-2}+x^2+8x+38$$

```
>> [b,a] = residue(r,p,k)      % 把上面的多项式转化为有理分式
b =
   1.0000    3.0000    4.0000     0    5.0000
a =
   1     -5     6
```

第三章 MATLAB 程序设计

MATLAB 有两种工作方式：一种是交互式的指令行工作方式，另一种是 M 文件的程序工作方式。在前一种工作方式下，用户只需在指令窗口中输入指令语句，MATLAB 就立即给出结果，这时 MATLAB 被当作一种高级数学演算纸和图形显示器来使用，第二章的所有实例均采用这种工作方式。对于需要大量指令行才能完成特定功能的情况，就需要用后一种工作方式。它实际上是把在 MATLAB 指令窗口下逐条输入的指令语句作为程序集中写入到一个文本文件中，这就是 M 文件。M 文件的扩展名一律为.m(M 文件的名称由此而来)。用任何字处理软件都可以对它进行编写和修改，最简单的方法是利用 MATLAB 本身的 M 文本编辑器。每当用户输入这个文件名时，文件中的指令就会由 MATLAB 执行。利用 MATLAB 进行编程，就是根据用户的不同需要，编写不同内容 M 文件的过程。

本章介绍进行程序设计的 M 文件的编制方法，包括指令文件与函数文件，然后介绍程序流程控制，包括循环结构和选择结构的控制方法。有了本章学到的编程方法，再结合前几章和第五章所学的内容，读者就可以自己编制“电路”、“信号与系统”等课程所需功能的程序了。

3.1 M 文件

3.1.1 M 文件编辑器

MATLAB 的文本编辑器是一个集编辑与调试两种功能于一体的工具环境，利用它不仅可以完成基本的文本编辑操作，还可以对 M 文件进行调试。

1. 启动编辑器的方法

创建新的 M 文件，启动编辑器有三种操作方法。

(1) 直接在指令窗口输入指令 edit 并按 Enter 键。

(2) 单击指令窗口工具栏上的□图标(New M - File)。

(3) 选择指令窗口的菜单项 File\New\M - File。

2. 打开已有 M 文件的三种操作方法

(1) 从图 3-1 所示的当前工作目录(Current Directory)窗口中选中要打开的 M 文件名并双击,或在选中的 M 文件名上单击鼠标右键,可弹出图 3-2 所示的快捷菜单,选择 Open 可以打开此 M 文件,选择 Run 可以运行此 M 文件。



图 3-1 当前工作目录窗口

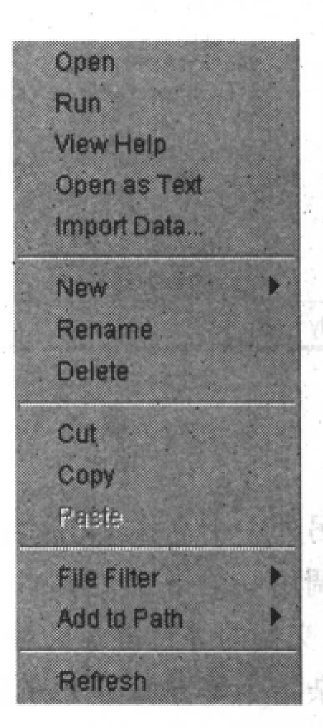


图 3-2 快捷菜单

(2) 单击指令窗口工具栏上的图标(Open File),再从弹出的对话框中选择需打开的 M 文件。

(3) 单击指令窗口中的菜单项 File\Open,再从弹出的对话框中选择需打开的 M 文件。图 3-3 是在 M 文本编辑器中打开的 M 文件。

3.1.2 指令文件与函数文件

在 MATLAB 中,M 文件有两类:指令(Script)文件和函数(Function)文件。这两类文件的命名必须以字母开头,其余部分可以是字母、数字或下划线(不能是汉字)。

1. 指令文件

指令文件的主要用途是使指令输入更加简单化。如果用户需要重复输入许多相同指令,即可将这些指令放在一个指令文件中(指令文件没有输入参数和输出参数)。可以说,指令文件就是将用户在 MATLAB 指令窗口中输入的一组

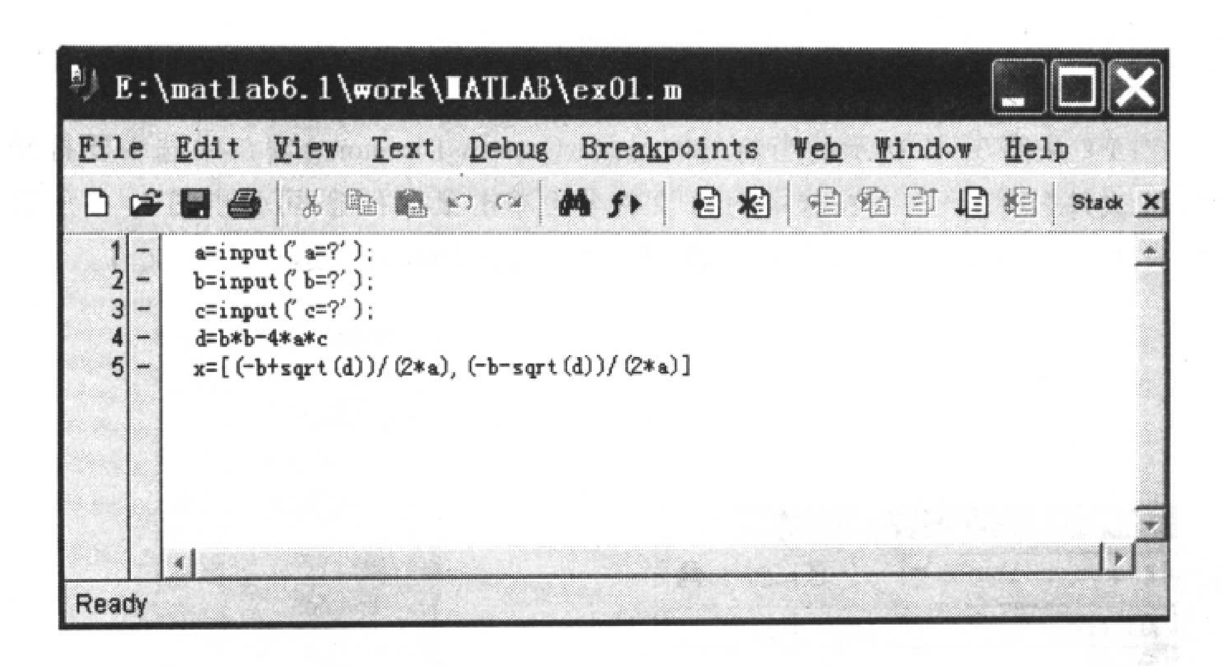


图 3-3 在 M 文本编辑器中打开的 M 文件

指令用另外一个名称(文件名)代替而已。

当调用一个指令文件时,MATLAB 自动执行文件中的一系列语句,在执行期间,并不交互的等待键盘的输入。因此,指令文件在实现分析问题、解决问题和设计复杂指令等方面是十分有用的。

在程序设计中,指令文件常作为主程序来设计。

指令文件的编写可以利用指令窗口的解释性执行功能,先在指令窗口逐条输入、执行,让系统自动检查错误,经修改确认正确后,再把这些指令行全部拷贝到 M 文件中。

文件编辑或修改好后,可单击编辑器工具栏的图标(Save),在弹出的对话框中输入文件名并保存。单击编辑器工具栏上的图标(Save and Run)也可以保存并运行该文件。如果有错误,则在指令窗口给出相应的提示,用户可根据出错提示找到出错的地方并进行修改。

2. 函数文件

函数文件的主要用途是扩充 MATLAB 的应用范围和满足用户不同的实际应用需求。函数文件可以接受输入变量,也可以返回输出变量。除了输入变量和输出变量以外,在函数文件内部的其他变量通常为该函数文件的局部变量,只在本函数的工作区内有效,一旦退出该函数,即为无效变量。它和指令窗口中的工作空间是相互独立的。而指令文件中定义或使用的变量都是全局变量,在退出文件后仍是有效变量,且被保留在工作空间中,其他指令文件和函数可以共享

这些变量。

函数文件必须以关键字 function 开头，第一行为函数说明语句，其格式为：

function[返回变元1,返回变元2,…] = 函数名(传入变元1,传入变元2,…)

其中，函数名由用户自己定义，其存储文件的文件名与函数名最好一致。若不一致，则在调用时应使用文件名。

用户可以通过函数说明语句中的返回变元和传入变元来实现函数变元的传递。返回变元和传入变元并不是必须的。下面是函数文件调用及变元传递的例子。

首先创建如下的函数文件并以 mean 为文件名保存。

```
function [m,s] = mean(a)
% 定义函数文件 mean.m,a 为传入变元,m 和 s 为返回变元
I = length(a);              % 计算传入向量的长度
s = sum(a);                 % 对传入向量 a 求和并赋值给返回变元 s
m = s/I;                    % 计算传入向量的平均值并赋值给返回变元 m
```

上述语句定义了一个新的函数 mean，其作用是对指定向量求和及平均值，并通过变元 s 和 m 返回计算结果。用户可以通过如下指令调用该函数：

```
a = 1:9;
[s,m] = mean(a)
```

按 Enter 键后的输出结果为

```
s = 5
m = 45
```

3.2 程序流程控制

3.2.1 数据的交互输入和输出

MATLAB 提供了一些输入输出函数，允许用户和计算机之间进行数据交换。

1. input 函数

如果用户想给计算机输入一个参数，则可以使用 input 函数来进行，该函数的调用格式为

```
a = input('提示信息','选项');
```

其中，提示信息可以为一个字符串，它用来提示用户输入什么样的数据。例如，用户想输入 a 矩阵，则可以采用下面的指令来完成：

```
a = input('Enter matrix a = >');
% 末尾加分号使输入的 a 矩阵不显示
```

按 Enter 键后，屏幕出现提示信息

```
Enter matrix a = >
```

然后等待用户从键盘按照 MATLAB 格式输入 a 矩阵。

如果在 input 函数调用时采用's'选项，则允许用户输入一个字符串。例如想输入一个人的姓名，可采用指令

```
xm = input('What''s your name:','s');
```

执行该语句后，在提示信息

```
What's your name:
```

后输入一个名字，这个字符串变量及内容就出现在工作空间中。

例如，在图 3-3 所示的 M 文本编辑器中编写指令文件，求一元二次方程 $ax^2+bx+c=0$ 的根。程序如下：

```
a = input('a = ? ');
b = input('b = ? ');
c = input('c = ? ');
d = b * b - 4 * a * c;
x = [(-b + sqrt(d))/(2 * a),(-b - sqrt(d))/(2 * a)]
```

保存该文件后，可以单击编辑器的运行图标执行该程序，也可以在指令窗口输入该文件名执行程序。这时在指令窗口出现的提示下输入如下参数：

```
a = ? 3
b = ? 4
c = ? 5
```

即可得到计算结果

```
x =
    -0.6667 + 1.1055i    -0.6667 - 1.1055i
```

再运行一次

```
a = ? 2
b = ? 6
c = ? 1
```

得

```
x =
    -0.1771    -2.8229
```

2. pause 函数

当程序运行时,为了查看程序的中间结果或者观看输出的图形,有时需要暂停程序的执行。这时可用 pause 函数,其调用格式为

pause(延迟秒数)

如果省略延迟时间,则将暂停程序,直到用户按任意键后才继续执行程序的后续语句。

3. disp 函数

MATLAB 提供的指令窗口输出主要有 disp 函数,其调用格式为

disp(输出项)

其中,输出项既可以是字符串,也可以是矩阵。

例如:

```
A ='Hello,MATLAB';
disp(A)
```

输出为

```
Hello,MATLAB
```

又如:

```
A =[1,2,3,1,2,3,1,2,2;4,5,6,4,5,6,4,5,6;7,8,9,7,8,9,7,8,9];
disp(A)
```

输出结果为

```
1   2   3   1   2   3   1   2   2
4   5   6   4   5   6   4   5   6
7   8   9   7   8   9   7   8   9
```

注意,和前面介绍的显示方式不同,用 disp 函数显示矩阵时将不显示矩阵的名字,而且其格式更紧凑,不留任何没有意义的空行。

3.2.2 条件语句

条件语句是根据给定的条件做出判断,分别执行不同的语句。MATLAB 提供的条件语句有 if 语句和 switch 语句。

1. if 语句

if 语句有两种格式。

格式 1:
```
if  逻辑表达式
    指令语句体
end
```

在程序执行该语句的过程中，首先判断逻辑表达式的值，如果逻辑表达式的值为真，那么程序就会执行指令语句体的所有语句；如果逻辑表达式的值为假，那么就跳过指令语句体，继续执行 end 语句之后的语句。注意这个 end 是决不可少的，没有它，当逻辑表达式的值为假时，就找不到继续执行的程序入口。

```
格式 2：if   逻辑表达式 1
               指令语句体 1
          elseif   逻辑表达式 2
               指令语句体 2
          …
          else
               指令语句体
          end
```

这里 elseif 和 else 语句都是可选的，只有 if 和 end 是必须的。

程序在执行过程中，首先判断逻辑表达式 1 的值，如果逻辑表达式 1 的值是真，就执行语句体 1。在语句体 1 执行完以后，就退出该结构，执行 end 后面的语句。如果逻辑表达式 1 的值是假，就继续判断逻辑表达式 2 的值。依此类推，如果 if 语句和所有 elseif 语句逻辑表达式的值都为假时，就执行语句体 else，然后执行 end 后面的语句。

例如判断正整数奇偶性的程序如下：

```
m = input('m = ');
if isempty(m) = = 1
     a = 'empty'
elseif rem(m,2) = = 0
     a = 'even'
else
     a = 'odd'
end
```

如果把 rem 函数改为 mod 函数，程序就可以判断任意整数的奇偶性，读者不妨一试。

2. switch 语句

switch 语句根据变量或表达式的取值不同，分别执行不同的语句。其格式为

switch 表达式（标量或字符串）

```
case 值1
        语句组 1
case 值2
        语句组 2
...
otherwise
        语句组
end
```

当表达式的值(或字符串)与某 case 语句中的值(或字符串)相同时,就执行该 case 语句后的语句组,然后直接跳到终点的 end。case 语句可以有多个。如果没有任何 case 的值与表达式的值相符,则将执行 otherwise 后面的语句组。

例如以下程序根据输入变量 m 的数值来决定要显示的内容。

```
m = input('请输入一个 10 以内的正整数');
switch m
case 1
     disp('I am a teacher.');
case 2
     disp('I am a student.');
case 3
     disp('I am a worker.');
otherwise
     disp('I am ');disp(m)
end
```

3.2.3 循环语句

循环语句是按照给定的条件,重复执行指定的语句。这是十分重要的一种程序结构。MATLAB 提供了两种循环语句:for 语句和 while 语句。

1. for 语句

for 语句通常用来执行循环次数已知的情况。它可以按照用户定义的次数来执行循环体中的内容。其调用格式为

```
for 循环变量 = 初值:步长:终值
   循环体语句
end
```

这里，初值、步长和终值可以是标量，也可以是表达式，步长为 1 时可以省略。当循环语句开始执行时，循环变量的值与初值相同。每执行一次循环体中的内容，循环变量的值就会按照步长的大小来改变。如果此数值介于初值和终值之间，则执行循环体语句，否则结束循环的执行，继续执行 end 后面的语句。

例如，已知 $y=1+\frac{1}{3}+\frac{1}{5}+\cdots+\frac{1}{2n-1}$，当 $n=100$ 时，求 y 的值。采用循环语句编写的程序如下：

```
y = 0;n = 100;
for i = 1:n
    y = y + 1/(2 * i - 1);
end,y
```

在实际编程中，采用循环语句会降低执行速度，所以上面的程序通常由下面的程序来代替：

```
n = 100;k = 1:2:2 * n - 1;y = sum(1./k);y
```

在这一程序中，首先生成一个向量 k，然后用 sum 函数求 k 向量各个元素的倒数之和。如果 n 由 100 改为 10 000，再分别运行这两个程序，可以明显看出后一种方法编写的程序比前一种方法快得多。

for 循环语句中的循环变量可以是一个向量。请看下例：

```
x = [1,2,3;4,5,6;7,8,9];        % 输入一个矩阵
for a = x                        % a 为列向量
    y = a(1) + a(2) + a(3)       % 求向量的元素和
end
```

该程序计算矩阵各列元素的和。

输出结果为

```
y = 12
y = 15
y = 18
```

2. while 循环语句

while 语句根据控制表达式来确定程序的运行方式。它一般用于事先不能确定循环次数的情况。语句格式为

```
while 控制表达式
    循环体语句
end
```

当表达式中的值为真时执行循环体。当循环体执行完毕后，继续判断表达式的值，如果仍为真就继续执行循环体。如此循环，直到表达式的值为假时终止循环。

下面给出一个用 while 语句实现求矩阵指数的例子。设 $\boldsymbol{A}$ 是给定的矩阵，根据幂级数展开式有

$$e^{A}=\boldsymbol{I}+\boldsymbol{A}+\frac{1}{2!}\boldsymbol{A}^{2}+\frac{1}{3!}\boldsymbol{A}^{3}+\cdots+\frac{1}{n!}\boldsymbol{A}^{n}+\cdots$$

设 E 是矩阵指数函数的值，F 是展开式的项，K 是项数，循环一直进行到 F 很小时为止。这里用到了范数函数 norm。程序如下：

```
A = rand(3);   % 定义一个方阵
E = zeros(size(A));   % 结果变量清零
F = eye(size(A));   % 展开式项初值
K = 1;   % 计算项数初值
while norm(F) > 0   % 用循环方式计算 A 的指数
  E = E + F;
  F = A * F/K;
  K = K + 1;
end
A
E
expm(A)   % 用函数方式计算 A 的指数
```

输出结果为

```
A =
    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214
E =
    3.1250    1.7453    1.1993
    0.6360    2.6358    0.1796
    1.8010    2.2981    2.6663
ans =
    3.1250    1.7453    1.1993
    0.6360    2.6358    0.1796
```

1.8010　　2.2981　　2.6663

运行结果表明,用幂级数展开法和用矩阵指数函数法得到的结果是一致的。

3. 循环的嵌套

如果一个循环结构的循环体又包括一个循环结构,就称为循环的嵌套。循环嵌套的实现仍是利用 for 语句和 while 语句。任意循环语句的循环体部分都可以包含另一个循环语句,且嵌套层数是任意的。根据嵌套层数分别称作二重循环、三重循环等。处于内部的循环为内循环,处于外部的循环为外循环。在设计多重循环时,要特别注意内、外循环之间的关系以及各语句放置的位置。

例如,求 100 ~ 1 000 之间全部素数的问题。素数是大于 1 且除了 1 和它本身以外不能被其他任何整数所整除的整数。因此,对于一个整数 m,用 2、3、4、5、…、$m-1$ 这些数逐个去除 m,看能否除尽。如果全部除不尽,则 m 是素数。程序如下:

```
n = 0;
for m = 100:1000
    flag = 1; j = m - 1;
    i = 2;
    while i < = j & flag
        if rem(m,i) = = 0
        % 用 2,3,4,…,m - 1 逐个除以 100 ~ 1000 之间的各数
            flag = 0; % 若能除尽,此数不是素数,flag 置零,结束 while 循环
        end
        i = i + 1;
    end
    if flag % 若不能除尽,flag 为 1,此数是素数
        n = n + 1;
        prime(n) = m;
    end
end
disp(prime)
```

程序中,外循环控制 m 的变化,内循环判断每一个 m 是否为素数。用变量 n 统计素数的个数,变量 prime 存放素数。

第四章 MATLAB 的符号运算

符号数学工具箱是操作和求解符号表达式的工具(函数)集合。它包括复合、简化、微分、积分以及求解代数方程和微分方程的工具,还有用于符号矩阵运算的工具,这样的运算不存在由数值计算引入的误差。

符号数学工具箱中的工具是建立在功能强大的 Maple 软件的基础上,它最初是由加拿大的滑铁卢大学开发的。当要求 MATLAB 进行符号运算时。它就请求 Maple 去计算并将结果返回到 MATLAB 指令窗口。因此,在 MATLAB 中的符号运算是 MATLAB 处理数字的自然扩展。

本章介绍 MATLAB 的符号运算方法,包括符号表达式和符号矩阵的创建,符号表达式和符号矩阵的运算,符号表达式的化简和展开,符号代数方程和符号微分方程的求解。

4.1 符号表达式和符号矩阵的创建

对于数值计算,不需要事先定义数值变量的类型和规模,但符号计算须事先说明计算中用到的变量是符号变量,否则,MATLAB 会把它默认为数值变量,按照数值计算的规则进行解释执行,结果出现错误信息。

MATLAB 在内部把符号表达式表示成字符串,以便和数字变量或运算相区别;否则这些符号表达式几乎完全就像基本的 MATLAB 指令。符号计算中出现的数字也是当作符号处理的。

符号表达式是代表数字、函数、算子和变量的 MATLAB 字符串或字符串向量,它不要求变量有预先确定的值;符号方程是含有等号的符号表达式。符号计算能使用户按熟悉的规则和给定符号恒等式求解这些符号方程。

1. 利用 sym 指令创建符号对象

定义基本符号对象的指令有两个:sym 和 syms。

```
a = sym('arg')
```

把数字、字符串、表达式 arg 转换成符号对象。

```
a = sym('arg', 'positive')        % 限定为正的符号变量
a = sym('arg', 'real')            % 限定为实符号变量
a = sym('arg', 'unreal')          % 复数符号变量
    syms arg1 arg2 ...
```

把多个变量定义为符号变量。注意各个变量名用空格分隔，而不是用逗号。

```
syms arg1 arg2 ... real           % 限定为正的符号变量
syms arg1 arg2 ... positive       % 限定为实符号变量
syms arg1 arg2 ... unreal         % 复数符号变量
```

例如要定义符号变量 x 和表达式 w，使用如下指令：

```
>>x = sym('x');w = sym('sqrt(2) - 1');
```

定义了符号变量或表达式后，即可进行计算。

```
>>f = w^2 + 1/w - 1
```

计算结果为

```
f =   (2^(1/2) - 1)^2 + 1/(2^(1/2) - 1) - 1
```

2. 字符串直接输入法创建符号对象

许多符号函数非常巧妙，能够将字符串转变为符号表达式。在 MATLAB 可以自己确定变量类型的场合下，为便于输入，不要求一定用显式函数 sym，可以用字符串定义。

例如：

```
>>f = 'sin(x)^2';
```

创建函数 $\sin^2(x)$ 并赋给变量 f，此时 f 可作为符号表达式使用。

```
>>eq = 'a * x^2 + b * x + c = 0';
```

创建方程 $ax^2 + bx + c = 0$ 并赋给变量 eq。

```
>>de = 'Dy + y^2 = 1';
```

创建微分方程 $\frac{dy}{dt} + y^2 = 1$。

```
>>diff('cos(x)');
```

求 $\cos(x)$ 对 x 的微分，其中符号表达式是用字符串形式提供的。

在某些情况下，尤其是建立符号矩阵时，必须用函数 sym 特别地将字符串变为符号表达式。

```
>> m = [a,b;c,d]     % 定义数值矩阵(a,b,c,d 的数值已知，否则出错)
>> n ='[a,b;c,d]'    % 定义字符串
n = [a,b;c,d]
```

```
>> o = sym('[a,b;c,d]')
% 定义符号矩阵(把字符矩阵转化为符号矩阵)
o =
      [ a, b]
      [ c, d]
```

在以上例子中,用 sym 定义和用字符串隐式定义矩阵的输出形式是不同的。

3. 把数值矩阵转化为符号矩阵

MATLAB 中的数值矩阵不能直接参与符号运算,必须通过转换才行。函数 sym 不仅可以把字符串转换为符号矩阵,也可以把数值矩阵转换为符号矩阵。不管数值矩阵的元素原先是用分数还是浮点数表示,转换后的符号矩阵都将以最接近的有理分式给出。例如:

```
>> m = [2/3,sqrt(3)/3,0.333;2.5,1/0.7,log(3)]     % 定义数值矩阵
m =
    0.6667    0.5774    0.3330
    2.5000    1.4286    1.0986
>> sym(m)      % 把数值矩阵转化为符号矩阵
ans =
      [        2/3,        sqrt(1/3),                    333/1000]
      [        5/2,        10/7,      4947709893870346*2^(-52)]
```

数值矩阵和字符串矩阵都可以在矩阵编辑器里修改,但矩阵编辑器无法修改符号矩阵。所以,对于较大规模且需要反复使用的符号矩阵最好建立一个 M 文件保存起来。

当字符表达式中含有一个以上变量而只有一个是独立变量,且用户未指明哪个是独立变量时,系统会自动使用默认的自变量。这是因为按照数学习惯,自变量一般为 x,y,z,t 等,在 MATLAB 中,也是按照这种数学习惯来确定表达式中的自变量。

符号表达式中默认的独立变量是唯一的。MATLAB 对单个英文小写字母(除 i、j 外)进行搜索,且以 x 为首选独立变量。如字符不是唯一的,就选择在字母顺序中最接近 x 的字母。如果有相连的字母,就选择在字母表中较后的那一个。

表 4-1 列出了一些表达式的系统默认独立变量。

表 4-1 表达式默认独立变量

表达式	默认独立变量
x^n	x
exp(u * y)	y
s * z + 2 * v	z
cos(w * t + θ)	t
h * theta^2	theta
sin((a + b) * alpha/2)	alpha
a * x1^3 + b * x1^2 - c	x1
i * (y2 - 2) + u^2 * v	y2

4.2 符号表达式和符号矩阵的运算

1. 符号四则运算和矩阵运算

由于 MATLAB 采用了重载技术，这使得用来构成符号计算表达式的运算符在名称和用法上，都与数值计算中的运算符几乎完全相同，这样极大地方便了用户编写程序。表 4-2 列出了常用的符号函数。直接使用与数值计算中名称完全相同的函数时，矩阵 A、B 和 a 必须是已经定义了的符号表达式或符号矩阵，而在阴影区的函数可以使用字符串变元。

表 4-2 符号矩阵运算符和函数

函数	说明	函数	说明
A + B symadd(A,B)	符号加	expm(a)	符号指数矩阵
A - B symsub(A,B)	符号减	eig(a) eigensys(a)	符号矩阵的特征值
A * B symmul(A,B)	符号乘	inv(a) inverse(a)	符号矩阵的逆
A/B symdiv(A,B)	符号除	poly(a) charpoly(a)	符号矩阵的特征多项式
rank(a)	符号矩阵的秩	jordan(a)	符号矩阵的约当标准型
diag(a)	符号矩阵的对角元素	det(a) determ(a)	符号矩阵的行列式

对两个已经定义了的符号表达式或符号矩阵 A,B 分别进行符号加、减、乘、除,可以直接使用符号 +、-、*、/运算,对于字符串矩阵 A,B 可以使用 symadd(A,B)、symsub(A,B)、symmul(A,B) 和 symdiv(A,B) 进行符号四则运算。例如:

```
>> a=sym('sin(t)');b=sym('cos(t)');a+b
% 事先定义符号表达式后再计算
ans = sin(t)+cos(t)
>> sym('x^2')/sym('log(y)')   % 符号表达式定义与计算同时进行
ans = x^2/log(y)
>> A='[a b;c d]';B='[x 1;y 2]';A*B
??? Error using ==> *
Inner matrix dimensions must agree.
>>A='[a b;c d]';B='[x 1;y 2]'; symmul(A,B)
ans =
      [ a*x+b*y, a+2*b]
      [ c*x+d*y, c+2*d]
```

可见,用"*"号计算字符串矩阵的乘积出错,若改用函数 symmul 计算,则会得到正确的结果。

与数值计算类似,符号计算也可以进行四则混合运算:

```
>> a=sym('sin(t)');b=sym('cos(t)');
>> c=sym('x^2');d=sym('log(y)');
>> e=a/b+c*d-1
e = sin(t)/cos(t)+x^2*log(y)-1
```

对于已定义的符号矩阵 a,一些矩阵运算函数也与数值计算的表示形式相同。如矩阵的秩、逆、行列式、指数等。例如:

```
>> rank(sym('[a b;c d]'))     % 计算符号矩阵的秩
ans = 2
>> n='[a,b;c,d]';m=sym(n);    % 定义字符串矩阵 n 和符号矩阵 m
>> determ(n)     % 计算字符串矩阵的行列式(不能用 det(n))
ans = a*d-b*c
>> det(m)     % 计算符号矩阵的行列式(也可以用 determ(m))
ans = a*d-b*c
>> m1=diag(m)     % 返回符号矩阵 m 的主对角线元素
```

```
m1 =
     [a]
     [d]
>>diag(m1)      % 返回符号向量 m1 对应的主对角线矩阵
ans =
     [a, 0]
     [0, d]
>> inv(m)      % 计算符号矩阵 m 的逆矩阵
ans =
     [d/(a*d-b*c),   -b/(a*d-b*c)]
     [-c/(a*d-b*c),   a/(a*d-b*c)]
>> poly(m)      % 计算符号矩阵 m 的特征多项式
ans = x^2-x*d-a*x+a*d-b*c
>> eig(m)      % 计算符号矩阵 m 的特征值
ans =
     [1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
     [1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
```

2. 符号微分

符号微分的函数是 diff,调用格式如下:

```
diff(f)          % 对默认的符号变量计算一阶微分
diff(f,x)        % 对指定的符号变量 x 计算一阶微分 df/dx
diff(f,x,k)      % 对指定的符号变量 x 计算 k 阶微分 d^kf/dx^k
```

例如,设函数 $f(x)=ax^4+bx^3-2cx+10$,对 x 求一阶导数:

```
>>f=sym('a*x^4+b*x^3-2*c*x+10');df=diff(f)
df=4*a*x^3+3*b*x^2-2*c
```

对 b 求一阶导数:

```
>> df=diff(f,b)
df = x^3
```

对 x 求三阶导数:

```
>> df=diff(f,x,3)
df = 24*a*x+6*b
```

又如,给定矩阵 $A=\begin{bmatrix} a^2 & bt^2 \\ x\sin t & \cos(tx) \end{bmatrix}$,求 $\frac{dA}{dx},\frac{dA}{dt},\frac{d^2A}{dtdx}$ 的语句如下:

```
>> syms a b x t,A = [a^2 b * t^2 * log(x);x * sin(t) cos(t * x)];
>> dax = diff(A)    % A 对 x 求一阶导数
dax =
       [        0,   b * t^2/x]
       [   sin(t),   - sin(t * x) * t]
>> dat = diff(A,t)       % A 对 t 求一阶导数
dat =
       [        0,        2 * b * t * log(x)]
       [   x * cos(t),    - sin(t * x) * x]
>> datx = diff(dat,x)       % A 对 t,x 求二阶混合偏导数
datx =
        [          0,             2 * b * t/x]
        [      cos(t),     - cos(t * x) * t * x - sin(t * x)]
```

3. 符号积分

符号积分也有不定积分、定积分和重积分等,与数值积分相比,符号积分指令简单,适应性较强,但是花费的时间较长。在 MATLAB 中,用函数 int 来求符号表达式的积分。调用格式如下:

```
intf = int(f,x)
% 对指定变量 x 求不定积分,x 缺省时对默认变量进行积分
intf = int(f,x,a,b)
% 对指定变量 x 求定积分,a 和 b 分别为积分的下限和上限
>> syms x z, v = int(x/(1 + z^2),z)    % 求不定积分
v = x * atan(z)
>> v1 = int(x * log(1 + x),0,1)    % 求定积分,上下限为常数
v1 = 1/4
>> syms x t, v2 = int(2 * x,sin(t),1)    % 求定积分,下限为符号表达式
v2 = 1 - sin(t)^2
>> syms x y,A = x^2 + y^2 + 1;    % 定义被积函数 A
>> B = int(int(A,y,x,x + 1),x,0,1)    % 求 A 的二重积分
B = 5/2
```

与符号微分相比,符号积分的难度要大得多。当函数的积分不存在时,系统将返回原来的积分表达式。

4. 求极限

MATLAB采用函数limit求极限，调用格式如下：

```
limit(f,x)
% 对f求x=0的极限，x缺省时，积分对系统默认的变量求极限
limit(f,x,a)          % 对指定表达式f求x→a的极限
limit(f,x,a,'left')   % 对指定表达式f求x→a的左极限
limit(f,x,a,'right')  % 对指定表达式f求x→a的右极限
```

例如：

```
>> syms x a t h,v1 = limit(sin(x)/x)
% 求函数sin(x)/x在x=0处的极限
v1 = 1
>> v2 = limit((sin(x+h) - sin(x))/h,h,0)    % 求函数sin(x)的导数
v2 = cos(x)
>> v3 = limit([(1+1/x)^x,exp(-x)],x,inf,'left')
% 求符号矩阵函数在x=+∞处的极限
v3 = [exp(1),       0]
```

5. 级数求和

级数求和的调用格式为

```
symsum(s,v,a,b)
```

表示对级数的符号通式s的变量v从v=a到v=b进行求和。若v省略，则对默认变量求和；若a,b省略，则对符号变量v从0到v-1求和。

例如，计算$1^2+2^2+\cdots+(k-1)^2$的调用格式为

```
>> syms k,symsum(k^2)
ans = 1/3 * k^3 - 1/2 * k^2 + 1/6 * k
```

又如，计算$\sum_{k=1}^{\infty}\frac{1}{k^2}$的调用格式为

```
>> syms k,symsum(1/k^2,1,inf)
ans = 1/6 * pi^2
```

6. 泰勒(Taylor)级数的展开

求泰勒展开式的调用格式为

```
taylor(f,n,a)
```

表示求符号变量(默认)在a处的n-1阶麦克劳林(Maclaurin)近似f的多项式。

```
>> syms x,taylor(log(x),6,1)
```

```
% 将 ln(x)在 x=1 处按 5 阶多项式展开
ans = x-1-1/2*(x-1)^2+1/3*(x-1)^3-1/4*(x-1)^4+1/5*(x-1)^5
```

4.3 符号表达式的化简和展开

化简表达式的目的是将符号表达式用最简明的字符表示。对于给定的表达式,应用表 4-3 列出的 MATLAB 函数可以推演出符合表达要求的形式。

表 4-3 表达式的符号化简

函 数	说 明	函 数	说 明
collect	合并同类项	expand	展开成多项式
factor	因式分解	horner	生成嵌套的多项式
numden	分离出分子和分母	simple	化简成最简形式
simplify	化简符号矩阵	subs	表达式中符号变量的替换

1. 表达式的化简

函数 simplify 利用函数规则(如三角函数恒等式)对表达式进行化简,调用格式为:

simplify(s)

函数 simple 是把表达式以最简的字符形式表示,并显示化简的过程。它通常与其他符号函数如 radsimp、combine、collect、factor、convert 等结合使用。调用格式为

```
simple(s)    % 显示所有可能的化简过程和化简结果
[f,how] = simple(s)    % f 给出化简结果,how 显示最佳化简过程
```

函数 collect 的作用是把符号表达式中相同幂次的项按照变量 v 进行合并。调用格式为

collect(s,v)

下面是几个例子。

```
>> syms x,simplify(cos(x)^2+sin(x)^2)    % 给出化简结果
ans = 1
>> [f,how] = simple(cos(x)^2+sin(x)^2)    % 给出最简结果及过程
f = 1
how = combine
```

说明:把表达式中的求和形式、乘积形式、幂形式各项合并

```
>> syms x y,collect(x^2 * y + y * x - x^2 - 2 * x,y)
% 按变量 y 进行同类项合并
ans = (x^2 + x) * y - x^2 - 2 * x
>> syms x y,collect(x^2 * y + y * x - x^2 - 2 * x)
% 按默认变量 x 进行同类项合并
ans = (y - 1) * x^2 + (y - 2) * x
```

2. 展开与分解

函数 expand 将表达式 s 中的元素按幂从高到低展开成多项式形式。函数 factor 把多项式 s 分解为多个因式,各因式的系数均为有理数。函数 horner 将符号多项式 s 用嵌套形式表示,即用多层括号的形式表示。调用格式为

```
expand(s)
factor(s)
horner(s)
>> expand(sym('(x + 1)^3'))
ans = x^3 + 3 * x^2 + 3 * x + 1
>> factor(sym('x^9 - 1'))
ans = (x - 1) * (x^2 + x + 1) * (x^6 + x^3 + 1)
>> horner(sym('x^3 - 6 * x^2 + 11 * x - 6'))
ans = -6 + (11 + ( -6 + x) * x) * x
```

3. 转换成分数

函数 numden 将给定的表达式 s 转化为用分数形式表示,其分子分母元素均用有理数表示。调用格式为

```
[n,d] = numden(s)
```

其中,n 表示分子,d 表示分母。例如:

```
>> [n,d] = numden(sym('x/y + y/x'))
% 计算 x/y + y/x = (x² + y²)/(yx)
n = x^2 + y^2
d = y * x
```

4. 变量替换

函数 subs 将用新变量(字符串或数字)替换表达式中的旧变量(必须是已定义的符号变量)。常用此函数把表达式转化为函数值。调用格式为

```
subs(s,'new',old)
```

```
% 用新字符串 new 替换原符号变量 old,得一新表达式
subs(s,new,old)
% 用数字 new 替换原符号变量 old,得表达式的值
例如:
>> syms a,subs(sin(1/3 * a * pi),'2',a)    % 结果是一个符号变量
ans = 1/2 * 3^(1/2)
>> syms a,subs(sin(1/3 * a * pi),2,a)      % 结果是一个数值变量
ans = 0.8660
```

4.4 方程求解

代数方程的求解指令是 solve,微分方程的求解指令是 dsolve,如表 4-4 所示。

表 4-4 方程求解函数

函 数	说 明	函 数	说 明
solve	求解代数方程	finverse	求反函数
compose	函数合成	dsolve	求解常微分方程

1. 代数方程求解

单一方程的求解指令为

solve(eq,var)

其中,函数 solve 用于求解代数方程 eq 中变量 var 的根。var 缺省情形下,采用系统默认的变量。方程 eq 应是一个单引号内的符号等式,若只是符号表达式(可不用单引号),MATLAB 默认表达式的值为 0。

代数方程组的求解指令为

solve(eq1,eq2,…,eqn,var1,var2,…,varn)

例如:

```
>> syms a b x,solve(a * x^2 - b * x - 6)
% 解方程 ax^2 - bx - 6 = 0,默认变量 x 有两个根
ans =
        [1/2/a * (b + (b^2 + 24 * a)^(1/2))]
        [1/2/a * (b - (b^2 + 24 * a)^(1/2))]
```

```
>> syms a b x y,[X,Y] = solve('a * x + b * y = 1 ','x - y = 2 ')
% 解方程组中默认变量 x 和 y 的根
X = (1 +2 * b)/(a + b)
Y = -(2 * a - 1)/(a + b)
>> syms x y t z
>>[st,sz] = solve('t * x + z * y - 1 = 0 ','2 * t * x - 3 * z * y + 2 = 0 ',z,t)
st = 1/5/x
sz = 4/5/y
```

上例中,根的顺序是按英文字母的顺序排列的。st 代表 t 的根,sz 代表 z 的根。

函数 compose 用于两个函数的复合运算,调用格式为

```
compose(f,g)
% 对函数 f(x)和 g(x)求复合函数 f(g(x))
```

例如:

```
>> syms x y ,f = 1/(1 +x^2);g = sin(y);compose(f,g)
ans = 1/(1 + sin(y)^2)
```

函数 finverse 用于求符号函数的反函数,调用格式为

```
finverse(f,u)
```

若变元 u 省略,以默认变量作为需变换的变量。

例如:

```
>> syms x y,f = x^2 + y;finverse(f,y)
ans =    - x^2 + y
```

2. 微分方程求解

函数 dsolve 用于求解常微分方程的符号解。调用格式为

```
r = dsolve('eq1 ,eq2 ,…','cond1 ,cond2 ,…','v')
r = dsolve('eq1 ','eq2 ',…,'cond1 ', 'cond2 ',…, 'v')
```

在 MATLAB 中,约定 D 表示一阶微分,D2 表示二阶微分,D3 表示三阶微分,…符号 Dy 相当于 dy/dt。函数 dsolve 把 D 后面的变量当作因变量,并且默认这些变量是对自变量 t 求导,也可以指定其他自变量 v。在使用该函数时,不能把 D 当作因变量。

微分方程的初始条件用单独的方程来表示,如果没有给出初始条件 cond,则系统认为是求微分方程的通解,在通解里包含积分常数 C1、C2、C3 等。

例如:

```
>> dsolve('Dy=1+y^2','y(0)=1')    % 解一阶微分方程
ans = tan(t+1/4*pi)
>> [X,Y]=dsolve('Dx=y,Dy=-x')    % 解微分方程组的通解
X = cos(t)*C1+sin(t)*C2
Y = -sin(t)*C1+cos(t)*C2
>> [f,g]=dsolve('Df=3*f+4*g,Dg=-4*f+3*g','f(0)=0,g(0)=1')
% 解微分方程组
f = exp(3*t)*sin(4*t)
g = exp(3*t)*cos(4*t)
>> dsolve('D2y=-a^2*y','y(0)=1,Dy(pi/a)=0')
% 解二阶微分方程
ans = cos(a*t)
```

第五章 MATLAB 的可视化功能

MATLAB 一向注重数据的图形表示，并不断地采用新技术改进和完善其可视化功能。作为一个优秀的科技软件，MATLAB 在数据可视化方面也有上乘表现。MATLAB 可以给出数据的二维、三维乃至四维的图形表示。通过对图形线型、立面、色彩、渲染、光线、视角等的控制，可把数据的特征表现的淋漓尽致。MATLAB 的图形功能很强，不但可以绘制一般函数的图像，而且可以绘制专业图形，如饼图、条形图等。

本章仅介绍 MATLAB 最基本的可视化功能，包括各种类型的二维图形的绘制和修饰，几种类型的三维图形的绘制。利用上述的绘图工具，读者就可以绘制电路、信号与系统等课程所需的图形和信号的波形了。

5.1 二维图形的绘制

MATLAB 提供了很多绘制二维图形的函数，它们以向量或矩阵作为输入变元来绘制图形。表 5－1 列出了其中常用的基本绘图函数。

表 5－1 基本绘图函数

函 数	功 能
plot	绘制直角坐标二维连续曲线图形
polar	绘制极坐标二维连续曲线图形
compass	绘制射线图
loglog	绘制对数图形（两个坐标轴都为对数坐标）
semilogx	半对数坐标图形（横轴为对数坐标，纵轴为线性坐标）
semilogy	半对数坐标图形（横轴为线性坐标，纵轴为对数坐标）
plotyy	绘制双纵轴图形
stem	绘制二维离散序列线型图

续表

函　数	功　　能
stairs	绘制阶梯图
fplot	绘制函数的曲线图
bar	垂直直方图
ezplot	绘制符号函数图形

1. 用 plot 函数绘制简单二维连续曲线

在二维曲线的绘图指令中，函数 plot 是最基本、最重要的二维图形绘图指令，其他许多绘图指令都是在它的基础上形成的。

plot 指令用来绘制直角坐标中的曲线。根据 plot 函数变元的不同，可以在平面上绘制不同的曲线。plot 函数是将各个数据点通过连折线的方式来绘制二维图形的。

调用格式 1：plot(y)

若 y 是向量，以 y 的序号作为横轴，按向量 y 的值绘制二维曲线。若 y 是矩阵，则按列绘制曲线图，曲线条数等于 y 矩阵的列数。

调用格式 2：plot(x,y)

这里 x 为横坐标，y 为纵坐标。若 x，y 是同规模的向量，则绘制一条曲线。若 x 是向量而 y 是矩阵，则绘制多条曲线，它们具有相同的横坐标数据。若 x，y 都是矩阵，则以它们对应的列构成二元组来绘制多条曲线。

调用格式 3：plot(x1,y1,'option',x2,y2,'option',…)

其作用为，以(x1,y1)，(x2,y2)为二元组，绘制多条曲线。每条曲线的属性有相应的选项 option 来确定。option 选项可以是表示曲线颜色的字符、表示线型格式的符号和表示数据点的标记，部分选项可以连在一起使用。曲线颜色、线型格式和标记如表 5-2 所示。

表 5-2　颜色和线型标记

符号	表示的颜色	符号	代表的线型	符号	代表的标记	符号	代表的标记
c	青色(cyan)	-	实线(缺省值)	+	加号标记	∧	△
m	品红(magenta)	--	虚线	o	圆圈标记	∨	▽
y	黄色(yellow)	:	点连线	*	星号标记	>	▷
r	红色(red)	-.	点划线	·	点标记	<	◁

续表

符号	表示的颜色	符号	代表的线型	符号	代表的标记	符号	代表的标记
g	绿色(green)	none	不画线	x	叉型标记	p	☆
b	蓝色(blue)			s	正方形标记	h	六角星符号
w	白色(white)			d	菱形标记		
k	黑色(black)						

例如：

```
>> x=0:pi/30:2*pi;y1=sin(x);y2=sin(x+pi/2);
% 定义曲线的横坐标 x 和纵坐标 y1、y2
>>plot(x,y1,x,y2,'rp')
% 绘制曲线 y1(x)、y2(x)
```

结果如图 5-1 所示。

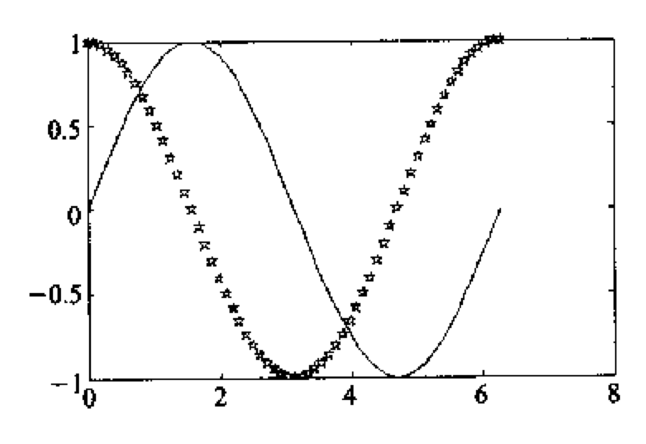


图 5-1 用 plot 函数绘制的正弦波形

2. 极坐标曲线的绘制

工程上有时需要绘制极坐标图形，MATLAB 用 polar 函数来绘制极坐标曲线。调用格式为

```
polar(theta,rho)
polar(theta,rho,'option')
```

以上格式表示以角度 theta 为一个坐标，单位是弧度，和另一个矢径 rho 坐标来绘制极坐标曲线。以 option 选项来确定曲线的属性，曲线属性的选取规则如表 5-2 所示。

例如：

```
>> t=0:01:2*pi; x=abs(sin(2*t).*cos(2*t));
% 定义角度坐标 t 和矢径坐标 x
```

```
>> polar(t,x)  % 绘制极坐标曲线
>> polar(t,x,'o')  % 绘制极坐标曲线,线型为实线圆圈
```

结果如图 5-2、5-3 所示。

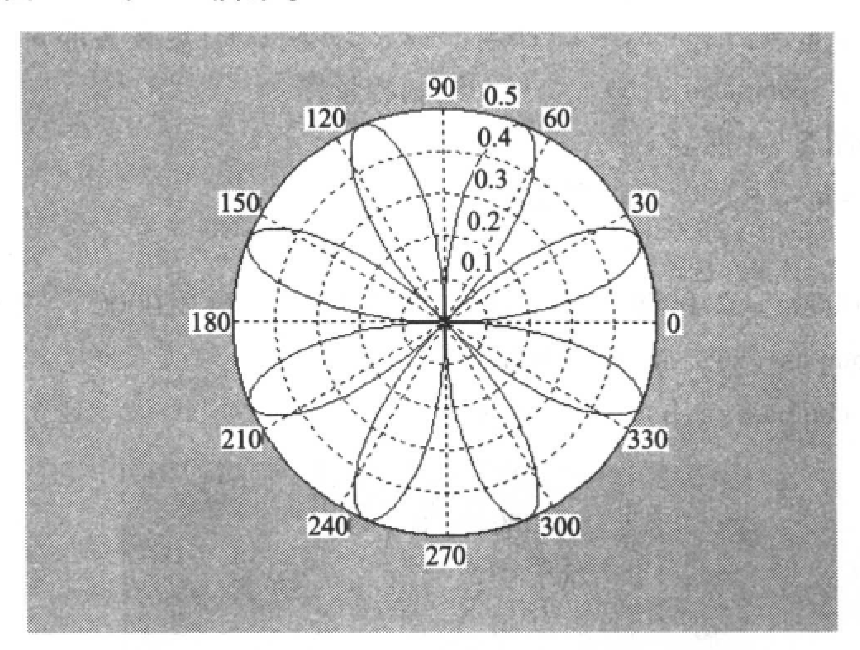


图 5-2 用 polar 绘制的极坐标曲线

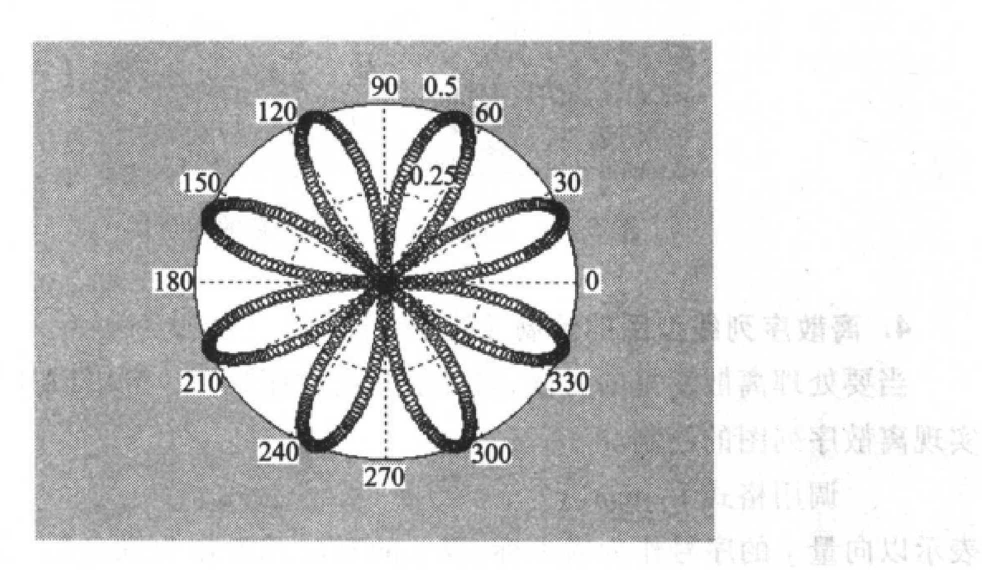


图 5-3 线型为实线圆圈的极坐标曲线

3. 射线图的绘制

在电路的正弦分析中常采用相量法,绘制相量图是一项基本的工作。MATLAB 用 compass 函数来实现相量图的绘制。调用格式为

```
compass(u,v)
compass(z)
```

表示绘制相量图。其中实向量 u 为相量的实部,实向量 v 为相量的虚部。复向量 z 为代数形式的复数,即:z = x + y * i 。

例如:

```
>>a =[1, -2,-2];b=[-2,3,-3]; % 定义实部 a 和虚部 b
>> compass(a,b)      % 绘制相量图
```

结果如图 5-4 所示。

```
>> z=a+i*b     % 定义复数矩阵 z
z =
    1.0000 - 2.0000i   -2.0000 + 3.0000i   -2.0000 - 3.0000i
>> compass(z)   % 绘制相量图
```

结果与 compass(a,b)一样也为图 5-4。

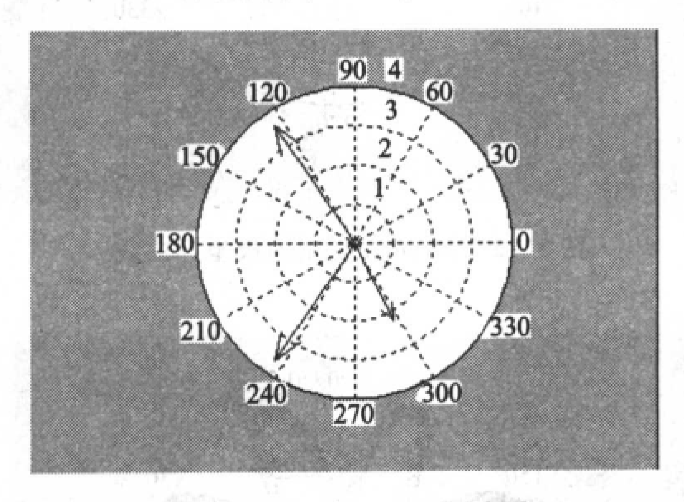


图 5-4 用 compass 函数绘制的相量图

4. 离散序列线型图的绘制(枝干图)

当要处理离散变量的时候,需要绘制离散序列图。MATLAB 用 stem 指令来实现离散序列图的绘制。

调用格式 1:stem(y)

表示以向量 y 的序号作为横坐标,以 y 的对应元素作为纵坐标,在(x,y)坐标点画一个空心小圆圈,并连接一条线段到横轴。

调用格式 2:stem(x,y,'option')

表示以向量 x 的各个元素为横坐标,以向量 y 的各个元素为纵坐标,在(x,y)坐标点画一个空心小圆圈,并连接一条线段到横轴。option 选项表示绘图时的线型、颜色,其取值见表 5-2。

调用格式 3:stem(x,y,'filled')

表示以向量 x 的各个元素为横坐标，以向量 y 的各个元素为纵坐标，在(x, y)坐标点画一个实心小圆圈，并连接一条线段到横轴。

```
>> x=0:0.1:4;y=sin(x.^2).*exp(-x);
>> stem(x,y)
>> stem(x,y,'filled')
```

绘制的图形见图 5-5 和图 5-6。

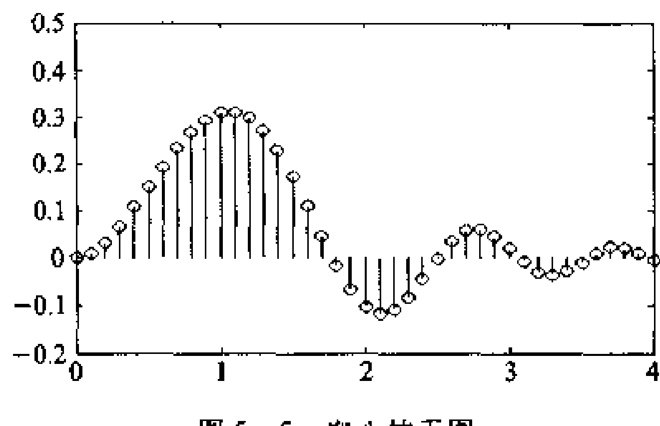


图 5-5 空心枝干图

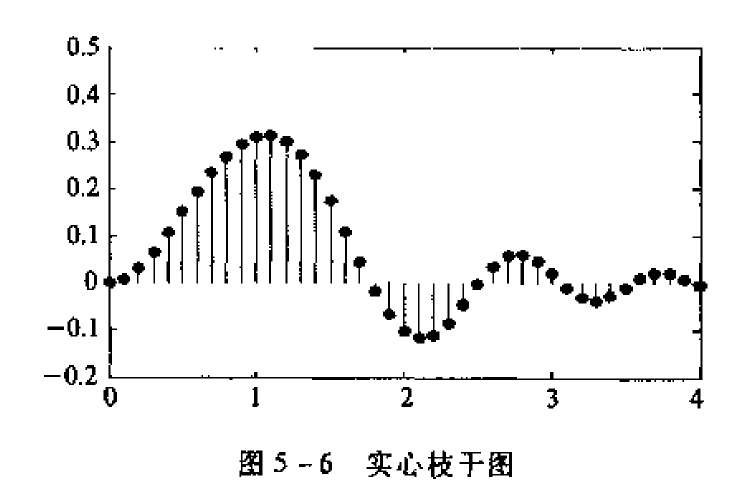


图 5-6 实心枝干图

5. 阶梯图形的绘制

对于有突变的分段常数波形，应选择采用阶梯图形。调用格式为

stairs(x,y)

它以 x 的间隔作为阶梯的宽度。

```
>> x=-2.5:0.25:2.5;y=exp(-x.*x);
>> stairs(x,y)      % 绘制指数函数的阶梯图形
```

结果如图 5-7 所示。

```
>> t=-0.1:0.1:3;x=((t>=0)+(t>=1)-2*(t>=2));
```

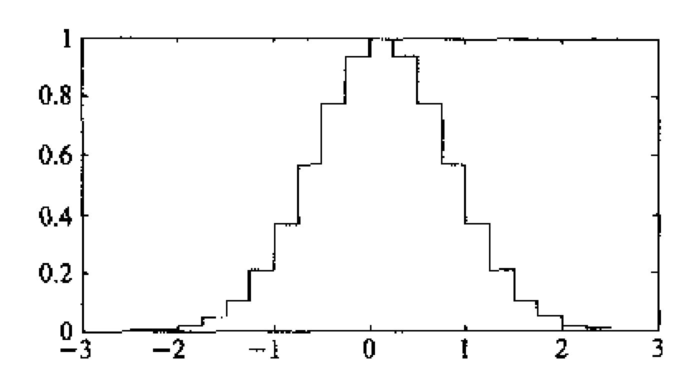


图 5-7 用 stairs 函数绘制的指数函数阶梯图形

```
>> stairs(t,x)
% 绘制 x = ε(t) + ε(t-1) - 2ε(t-2)的阶梯图形
>> plot(t,x)
% 用 plot 函数绘制同一波形突变效果会变差
```

结果如图 5-8、5-9 所示。

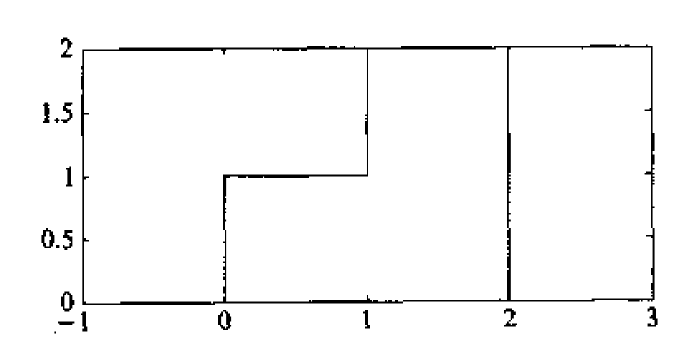


图 5-8 用 stairs 函数绘制的波形

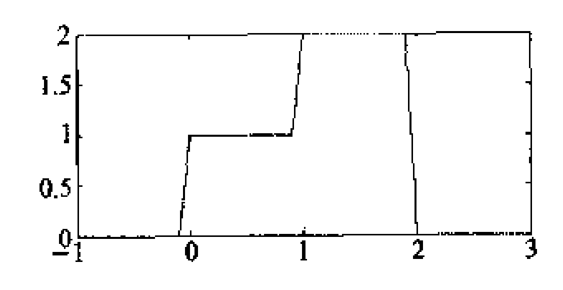


图 5-9 用 plot 函数绘制的波形

6. 符号函数图形的绘制

对于用符号和字符串表示的函数 $y = f(x)$,MATLAB 也提供了绘图指令 *ezplot*。调用格式为

```
ezplot(F,[xmin,xmax],fig)
```

其中，符号变量 F(也可以是字符串变量)是待绘制图形的符号函数；[xmin，xmax]是界定绘图的自变量范围，缺省时，系统默认值为[-2π,2π]；fig 是指定的图形窗口，缺省时为当前图形窗口。图形中会自动加注自变量和因变量的符号。例如：

```
>> y = sym('cos(10 * t) * sin(t)');
>> ezplot(y)
```

绘制的图形见图 5-10。

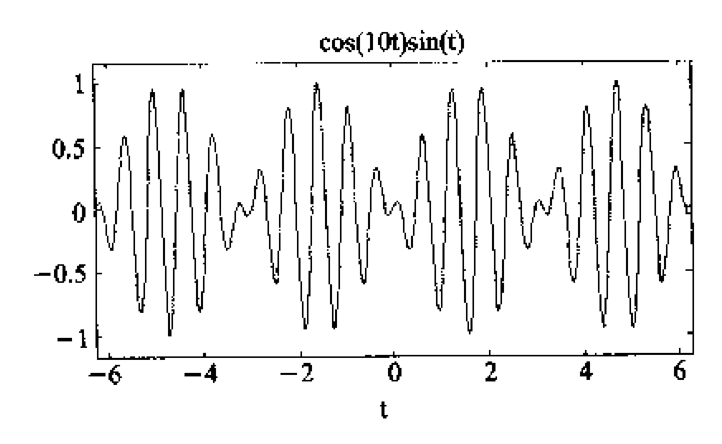


图 5-10 用 ezplot 函数绘制的波形

5.2 二维图形的修饰

5.2.1 利用 MATLAB 指令修饰图形

在利用 5.1 节介绍的函数绘图时，MATLAB 按默认的设置及用户指定的数据绘制图形。MATLAB 还提供了一些图形函数，专门用于对绘图指令所绘出的图形进行修饰。常见的图形修饰函数如表 5-3 所示。

表 5-3 图形修饰函数

图形修饰函数	说　明	图形修饰函数	说　明
axis	改变坐标轴状态	box	坐标框开关
grid	网格线开关	gtext	用鼠标标注文字
hold	图形刷新开关	legend	图例标注
line	加绘折线	subplot	图形窗口分割
text	加文本标注	title	加图形标题
xlabel	标注 x 轴名称	ylabel	标注 y 轴名称

1. 坐标状态的控制

(1) 改变坐标轴状态 axis。

MATLAB 可以自动地根据曲线数据的范围选择合适的坐标系，从而使得曲线尽可能清晰地显示出来，所以在一般情况下不必专门对坐标系进行设置。但是，如果对 MATLAB 自动产生的坐标轴不满意，可以利用 axis 指令对坐标轴进行调整。调用格式为

```
axis([xmin,xmax,ymin,ymax])
```

其中，xmin、xmax、ymin 和 ymax 分别表示横轴和纵轴的最小值和最大值。在调用该指令时要确保最大值大于最小值。其他调用格式为

```
axis square    % 使横轴和纵轴的长度相同
axis equal     % 使横轴和纵轴的刻度单位相同
axis tight     % 使坐标轴的区域和图形的区域正号吻合
```

例如：

```
>> t=0:1/100:2*pi;x=2*sin(t);y=cos(t);plot(x,y)
% 系统自动确定坐标轴状态(图 5-11)
>> axis square    % 把当前图形设置为正方形图形(图 5-12)
>> axis equal     % 把当前图形的坐标轴刻度设置为相等(图 5-13)
>> axis tight     % 把当前图形设置为占满坐标轴区域(图 5-14)
```

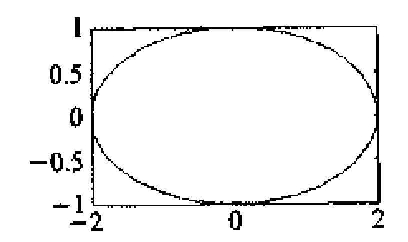


图 5-11 MATLAB 默认的坐标宽度

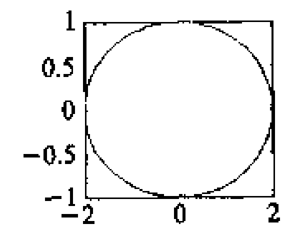


图 5-12 横轴与纵轴长度相同

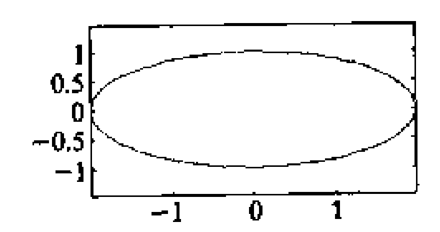


图 5-13 横轴与纵轴刻度单位相同

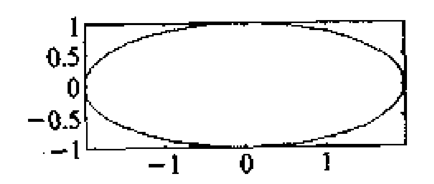


图 5-14 图形占满坐标轴区域

(2) 坐标框开关 box。

在默认状态下,系统自动用一个坐标框把图形圈起来。如果只需画出坐标轴,可以利用 box 开关进行控制,其指令为

```
box on          % 添加坐标框
box off         % 删去坐标框
```

如已画出的图 5-11 为当前图形时,执行指令

```
>> box off
```

即可得到如图 5-15 所示的没有坐标框的图形。

(3) 添加坐标轴 line。

MATLAB 自动把坐标轴画在边框上,如果需要从坐标原点拉出坐标轴,可以利用 line 指令,它用于在图形窗口的任意位置画直线或折线,其指令为

```
line(x,y)
```

此指令表示在当前图形窗口中绘制一条由向量 x 和 y 的对应元素为数据点的折线。

如已画出的图 5-2 为当前图形时,执行指令

```
>> line([0 4],[0 0])
```

即可在图形中从原点拉出一条直线作为横坐标,如图 5-16 所示。

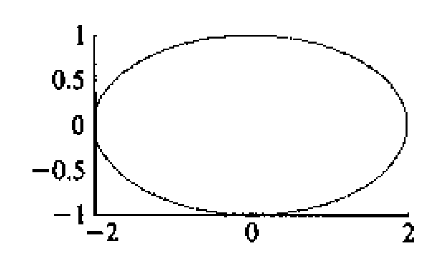


图 5-15 无坐标框的图形

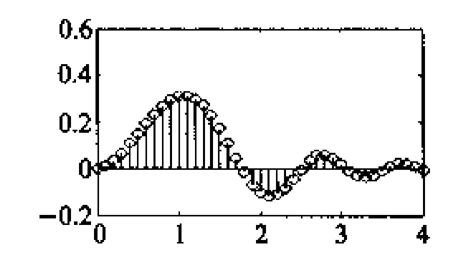


图 5-16 从原点拉出横坐标

2. 图形的标注和图例

为了使图形的信息更加清晰丰富,往往需要在图形的合适位置加注文字说明。MATLAB 提供了许多图形的标注指令。

(1) 标注坐标轴。

```
xlabel('string')    % 在 x 轴合适位置标注 x 轴的标题,string 为字符串
ylabel('string')    % 在 y 轴合适位置标注 y 轴的标题,string 为字符串
```

(2) 加注图形标题。

```
title('string')     % 在图形的上方标注图形的标题,string 为字符串
```

(3) 加注文本说明。

text(x,y,'string')　　% 在指定位置(x,y)处加入字符串

gtext('string')　　% 利用鼠标定位,在图形中加入字符串

(4) 图例标注。

legend('string1','string2',…)　　% 给图形加入图例

使用该指令可对当前图形建立一个图例说明盒,盒内给出用户指定的图例说明。此外,利用指令

legend off

可以移去图例说明盒。

下列语句绘制了三条曲线并添加各种标注,其结果如图 5-17 所示。

```
>> x = linspace(0,2 * pi,50);y = sin(x);z = cos(x);w = 0.25 * x - 0.5;
% 定义三个函数 y(x)、z(x)、w(x)
>> plot(x,y,'b:',x,z,'r',x,w,'*');
% 用不同线型绘制 y(x)、z(x)、w(x)的曲线
>> axis([0,2 * pi, -1.2,2]);   % 确定横轴和纵轴的范围
>> xlabel('x axis');ylabel('function y,z,w'); % 标注横轴和纵轴的变量
>> title('Three functions:y,z,w');   % 在上方标注图形的标题
>> text(5.5, -0.8,'y = sin(x)');   text(2.2, -0.5,'z = cos(x)');
% 在两曲线的指定位置标注函数式
>> gtext('w = 0.25x - 0.5');      % 利用鼠标定位标注 w 曲线的函数式
>> legend('y','z','w')      % 标注三条曲线的图例
```

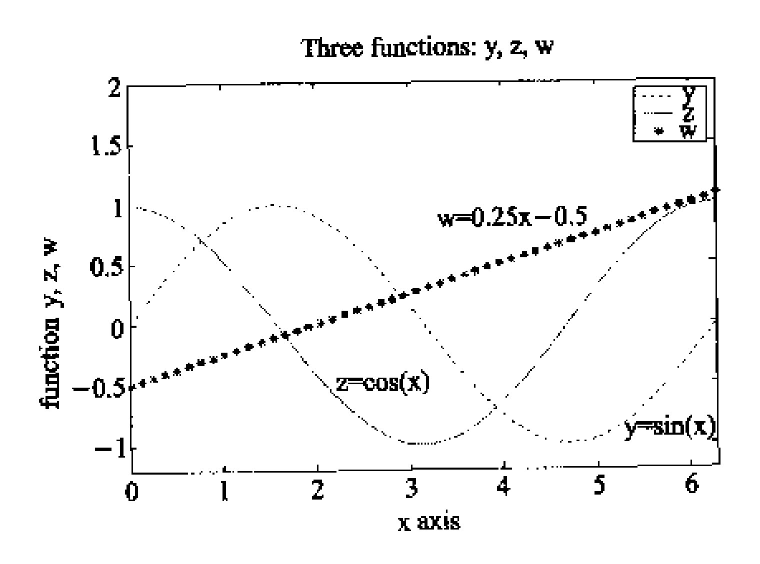


图 5-17 给图形添加多种标注

3. 图形窗口的控制与表现

(1) 图形的重叠控制。

在 MATLAB 中可以在同一坐标系中绘出多幅图形。利用 plot 指令虽然也可以做到这一点,但它在执行时首先将当前图形窗口清屏,用户只能看到最后一条 plot 指令绘制的图形。MATLAB 提供了 hold 指令,将当前窗口的图形保留,利用多条 plot 指令绘制多幅图形。

```
hold on          % 保留当前窗口的图形
hold off         % 解除 hold on 指令
```

(2) 坐标网格的控制。

```
grid on          % 添加坐标网格
grid off         % 解除 grid on 指令
```

(3) 图形窗口的分割。

MATLAB 允许一个图形窗口中显示多个图形,此功能可利用 subplot 指令实现。

```
subplot(m,n,p)
```

表示将当前图形窗口分割为 m × n 个绘图子区域,并将第 p 个子区域作为当前的绘图区域。在图形区域的号码编排中,采用行优先的原则,即:所有的子图沿着第一行从左往右编号,然后第二行,依此类推。

下例把一个图形窗口分割为四个图形区域,并利用各种控制函数加以修饰,其效果如图 5 - 18 所示。

```
t = -10:0.05:10;x11 = cos(t);x12 = t;x2 = t.*cos(t);
x3 = t.*sin(t);x4 = sin(t).*cos(t);
subplot(2,2,1);plot(t,x11);
hold on; plot(t,x12);   % 在第一个坐标系绘制两条曲线
title('x11 = cos(t) and x12 = t');
grid on;     % 标注第一个坐标系内曲线的标题并加网格线
subplot(2,2,2);plot(t,x2);
title('x2 = t*cos(t)');grid on;   % 绘制第二个坐标系内的曲线
subplot(2,2,3);plot(t,x3);
title('x3 = t*sin(t)');   % 绘制第三个坐标系内的曲线
subplot(2,2,4);plot(t,x4);
title('x4 = sin(t)cos(t)');grid on   % 绘制第四个坐标系内的曲线
```

在一个图形窗口中,绘制的多个子图之间是相互独立的,若单独对某一个子图进行修改和标注等操作,并不会影响其他的子图。对子图的操作就是对当

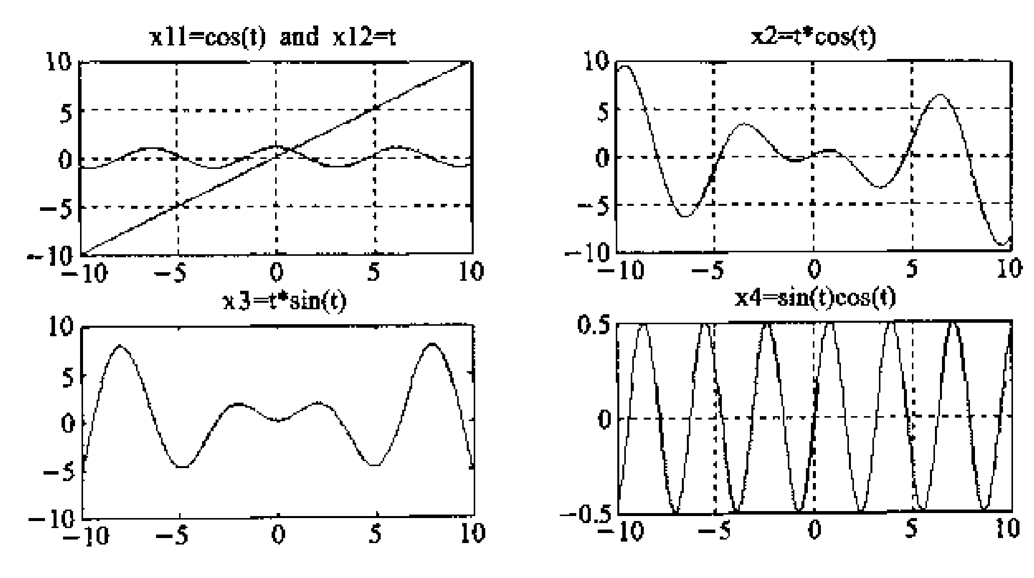


图 5-18 图形控制函数 hold、grid、subplot 的应用效果

前子图进行操作。如果用户想修改某一子图,必须首先使这个子图成为当前子图。每个图形窗口中都只有一个当前子图,系统默认的当前子图是最后使用的那个子图。如果用户想把上例图中的第三个子图作为当前子图,方法为:(1) 用鼠标单击图形窗口的第三个子图;(2)运行函数 subplot(2,2,3)。

如果函数 subplot 中 m 和 n 的数目与原来不同,那么 MATLAB 将会按照新的子图分割指令来改变图形窗口中的子图号码,并且将原先的子图清除,为新的子图腾出空间。

如果要回到图形窗口内只有一个子图的默认状态,可以用 subplot(1,1,1) 指令实现。

5.2.2 利用图形窗口的交互功能修饰图形

利用图形的输出窗口菜单对绘出的图形进行交互修饰与控制,是 MATLAB6.0 新增的功能。它让用户对图形的一般属性控制变得极为简单,使用户无需记住图形的修饰控制指令,就能轻而易举地达到控制的目的,从而把用户从对一大堆繁杂的指令的记忆中解放出来。

例如,先利用如下指令绘制一个简单图形。

```
>> x=0:0.1:7;y1=2*sin(x);y2=sin(2*x);stem(x,y1+y2);
>> hold on;plot(x,y1,x,y2)
```

在 MATLAB 图形窗口的顶部有一个菜单栏。该菜单栏有七个选项,单击任意一个菜单选项,就会弹出一个下拉式菜单,这里有各种指令选项。

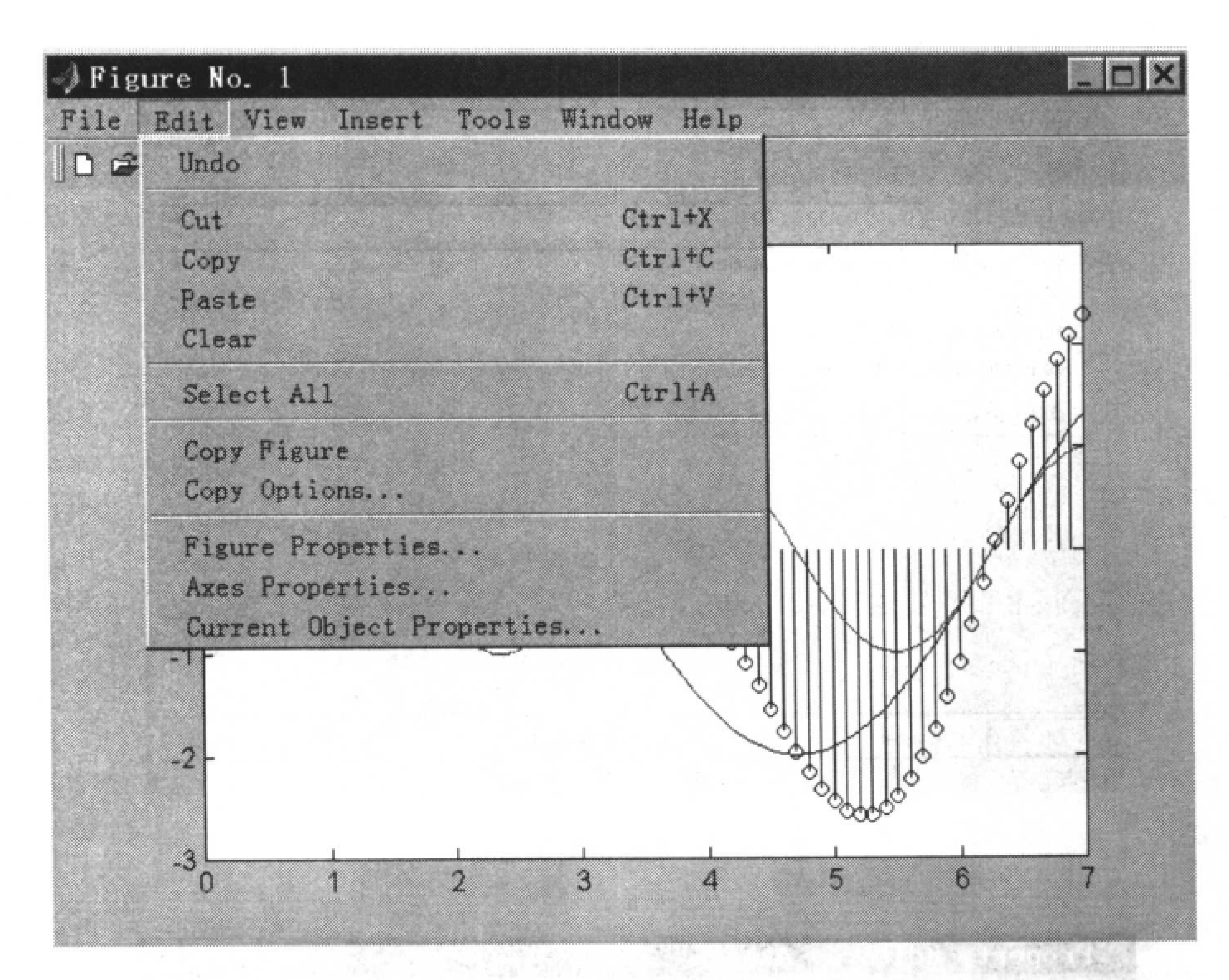


图 5-19 图形窗口的菜单栏及 Edit 下拉式菜单

1. 使用 Edit 编辑菜单项交互修饰图形

在菜单栏内单击 Edit 选项，即可看到编辑菜单中的各种选项指令，如图 5-19 所示。

Edit 下拉菜单的底部有三个子项，它们分别为

(1) Figure Properties…：图形窗口的属性。

(2) Axes Properties…：坐标轴的属性。

(3) Current Object Properties…：当前对象属性。

单击上面任一个选项，会打开一个对应的属性编辑窗口。图 5-20 是打开的 Figure 编辑窗口，图 5-21 是打开的 Axes 编辑窗口。

在 Figure 编辑窗口中，可以对图形窗口的 Style(显示风格)和 Title(标题)作改动。例如在 Style 选项卡中把 Background color(背景颜色)栏选为 White(白色)；在 Title 选项卡中 Figure window name(图形窗口名)栏内填入：交互控制举例。还可以从 Figure 编辑窗口的上部 Edit Properties for 栏内的下拉框中选择欲编辑的对象，图 5-22 选中的是用枝干图绘制的 y1 + y2 图形的顶部标记，单击

图 5－20　Figure 编辑窗口

图 5－21　Axes 编辑窗口

最下面的 line 对象，会弹出图 5-23 所示的 line(曲线)编辑窗口，在这里把 Face color 栏改为 blue，把 Size 栏改为 4.0。照此方法也可以选择其他的曲线进行修饰。修改后的效果见图 5-24。另外，在图形窗口中双击欲编辑的对象也可以打开相应的对象编辑窗口。

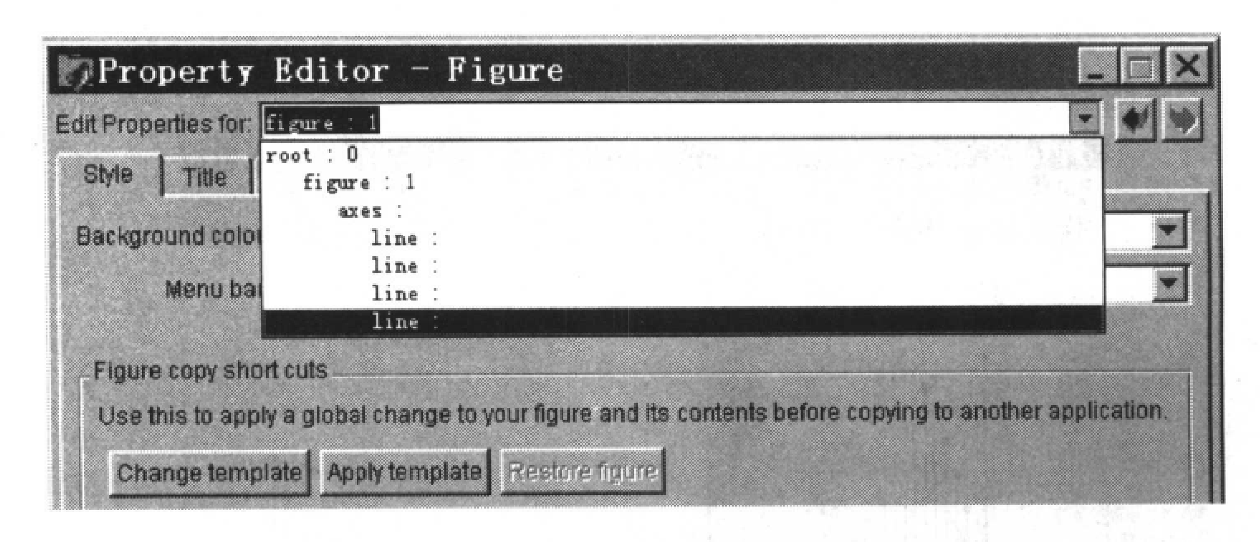


图 5-22 选取编辑对象

图 5-23 编辑图形曲线

在 Axes 编辑窗口可以对坐标轴的属性进行编辑，例如，在 Scale 选项卡中控制各个坐标轴的显示范围、标注方式、网格是否显示等；在 Style 选项卡中控制各个坐标轴的线宽、颜色、位置、图框是否显示、坐标标注字体等。例如把 Axis box on 前复选框的对钩去掉，即可取消默认的坐标框，如图 5－24 所示。

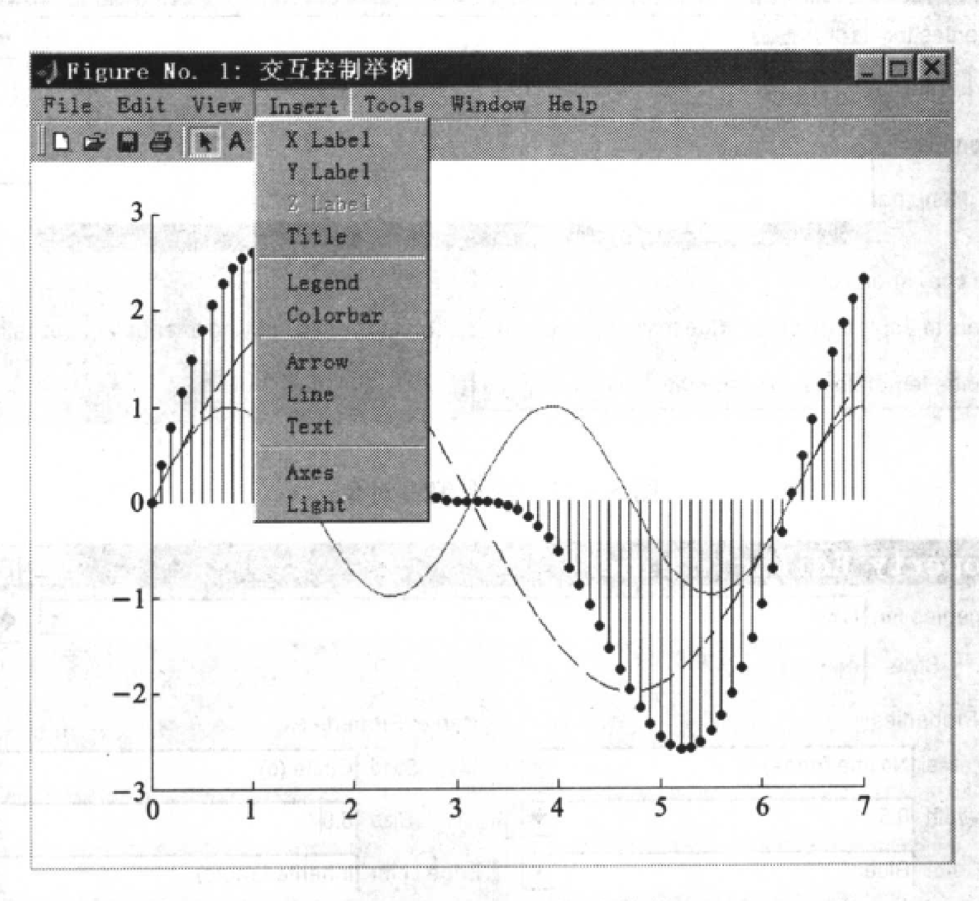


图 5－24　Insert 下拉式菜单

2. 使用 Insert 编辑菜单项交互修饰图形

单击菜单栏的 Insert 选项，会出现如图 5－24 所示的下拉菜单。在这里可以插入或修改坐标轴标记、图形的标题、图例说明、直线、箭头、文本说明、光照效果等。例如，在图 5－24 所示的图形窗口中插入从原点引出的横坐标轴、图例说明、坐标轴标记等，效果如图 5－25 所示。

如果需要对插入的对象进行修改和移动位置，直接双击它即可进行操作。

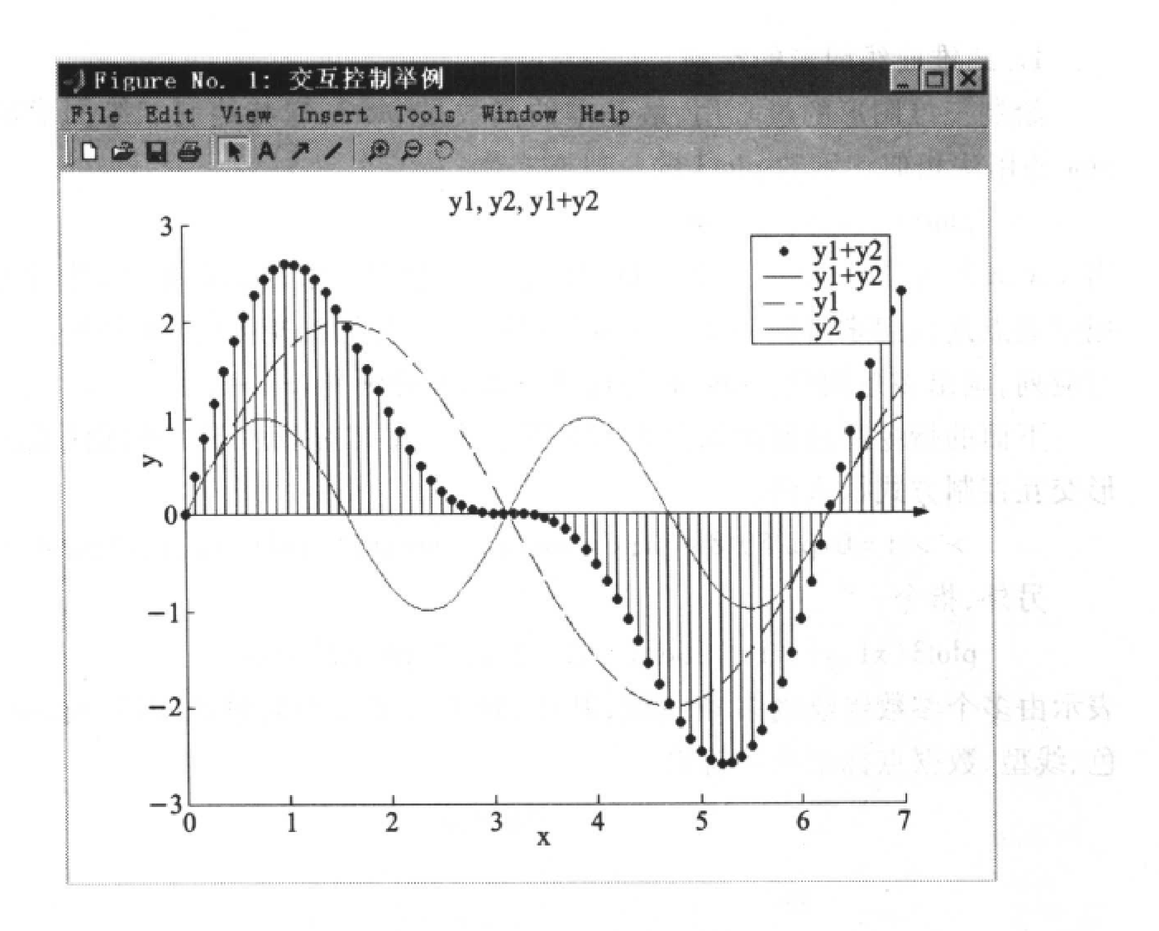


图 5-25 交互控制后的效果

5.3 三维图形的绘制

MATLAB 提供了许多实现三维数据可视化的函数,可以绘制三维空间曲线或曲面。常用的三维图形函数见表 5-4。

表 5-4 三维图形函数

图形函数	说明	图形函数	说明
plot3	绘制三维曲线	mesh	绘制网格曲面
meshc	等高线网格曲面	meshz	带垂帘的网格曲面
surf	绘制曲面图	surfc	等高线曲面图
surfl	带光照的曲面图	surnorm	带法线的曲面图
shading	清除网格线	meshgrid	产生栅格数据矩阵
pie3	三维饼图	bar3	三维垂直直方图
stem3	三维枝干图	slice	切片图

1. 三维曲线图形的绘制

绘制三维图形的最常用、最基本的函数是 plot3，它与绘制二维图形的函数 plot 的用法相似。函数 plot3 的一般格式为

```
plot3(x,y,z,'option')
```

当 x、y、z 为向量时，将以三个向量中的相应元素坐标绘制出数据点，然后再用线把这些点连接起来得到一条空间曲线；当 x、y、z 为同维矩阵时，则分别取它们的对应列，画出多条曲线。option 为选项参数，其规则见表 5－2。

下面的指令将绘制出如图 5－26 所示的三维螺旋线。图中的标注是利用图形交互控制方式插入的。

```
>> t=0:pi/50:8*pi;x=sin(t);y=cos(t);plot3(x,y,t);grid on
```

另外，指令

```
plot3(x1,y1,z1,'ption1',x2,y2,z2,'option2',…)
```

表示由多个参数组绘制多条曲线，其中，每个 x，y，z 为向量或矩阵，option 为颜色、线型、数据点标记的字符串。

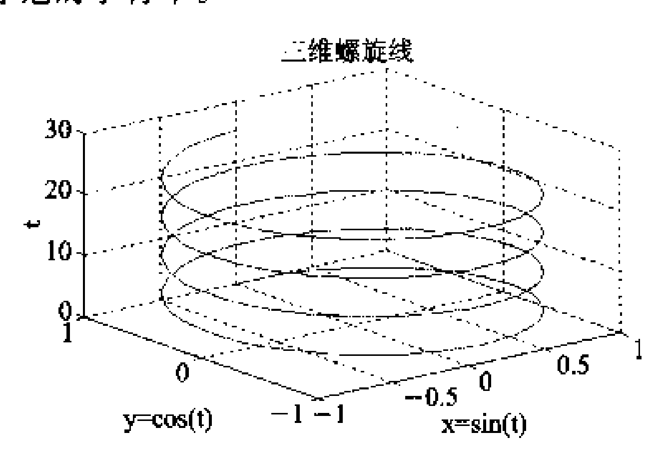


图 5－26 绘制的三维螺旋线

2. 三维网格曲面的绘制

在 MATLAB 中，三维网格曲面图所构成的网格状表面由 x－y 平面上的矩形栅格及相应的 z 坐标构成，相邻点之间用直线连接。对于显示大型数据矩阵或双变量的函数是很有用的。

在绘制网格曲面之前，需要知道各个四边形顶点的三维坐标值。一般应先确定二维坐标(x,y)，然后利用函数式计算出 z 坐标。二维坐标是一种栅格形的数据点，可由 meshgrid 指令产生。其调用格式为

```
[X,Y]=meshgrid(x,y)
```

表示由向量 x 和 y 通过复制的方法产生绘制图形时所需的栅格数据 X 矩阵和 Y

矩阵。该函数产生栅格数据的方法是：将向量 x 作为矩阵 X 的一个行向量，并将它复制 length(y)次，以构成栅格数据点 X 矩阵；同样，将向量 y 作为矩阵 Y 的一个列向量，并将它复制 length(x)次，以构成栅格数据点的 Y 矩阵。

例如，下面的指令将产生 X 和 Y 矩阵。

```
>> x=[1,2,3,4,5];y=[7,8,9];[X,Y]=meshgrid(x,y)
X =
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
Y =
     7     7     7     7     7
     8     8     8     8     8
     9     9     9     9     9
```

mesh 函数用于创建矩阵数据的三维网格图形。调用格式为

mesh(X,Y,Z)

表示在三维空间中画出一个彩色的、带有线框的表面视图，MATLAB 同时将该视图在三维空间中显示出来。参数 X、Y、Z 都是矩阵，X 矩阵的行向量相同，Y 矩阵的列向量相同。

例如，对于函数 $f(x,y)=\dfrac{\sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}$，下面的指令可以绘制出如图 5-27 所示的网格曲面图形。

```
>> x=-8:0.5:8;y=x;
>>[X,Y]=meshgrid(x,y);R=sqrt(X.^2+Y.^2)+eps;
>> Z=sin(R)./R;mesh(X,Y,Z);grid on
```

3. 三维阴影曲面的绘制

函数 surf 用于创建矩阵数据的三维阴影曲面图形。调用格式为

surf(x,y,z)

该格式将创建一个彩色的、由多个小面组成的表面视图，MATLAB 同时将该视图在三维空间中显示出来。通常，这些小面是四角形的，每个都有固定的颜色，而边界是黑色的网格线。

shading 函数用来清除网格线。调用格式为

shading flat

在绘制二元函数的图形时，需要首先创建函数定义域中的对应矩阵 X 和矩

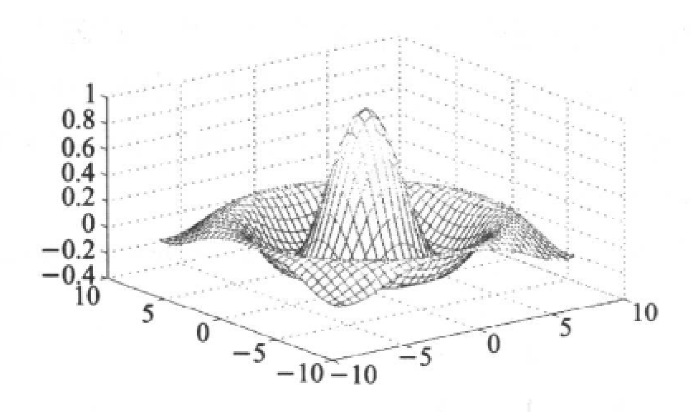


图 5 - 27 绘制的三维网格曲面

阵 Y,然后再利用这两个矩阵计算和绘制该二元函数。一般是先用函数 meshgrid 创建矩阵 X 和矩阵 Y,利用函数 meshgrid 得到的数据点是均匀分布的,然后可以继续用函数 mesh 和 surf 等绘制图形。

对于上面的例子,用 surf 函数绘制图 5 - 28 所示彩色曲面图的指令为

```
>> x = -8:0.5:8;y = x;
>>[X,Y] = meshgrid(x,y);R = sqrt(X.^2 + Y.^2) + eps;
>> Z = sin(R)./R;surf(X,Y,Z);grid on
```

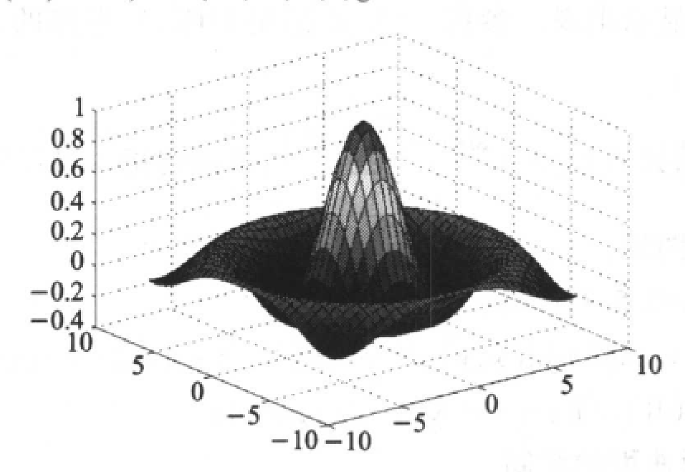


图 5 - 28 带网格线的曲面图

如果再加上一条指令

```
>> shading flat
```

可得到图 5 - 29 所示的去掉网格线的曲面图。注意,阴影曲面图与网格曲面图是不同的。阴影曲面图的网格线条是黑色,线条间的曲面由各种彩色填充;而网格曲面图的网格线条是各种彩色的,线条间的曲面是无色的。

对于三维图形的修饰,可以仿照二维图形的图形交互控制的方法。其他三

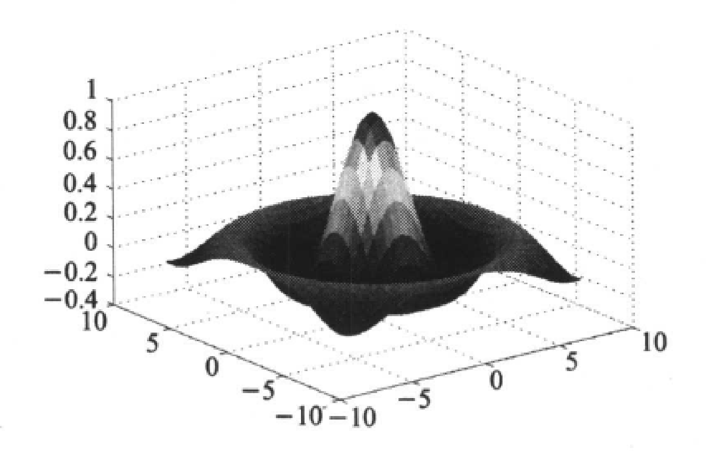
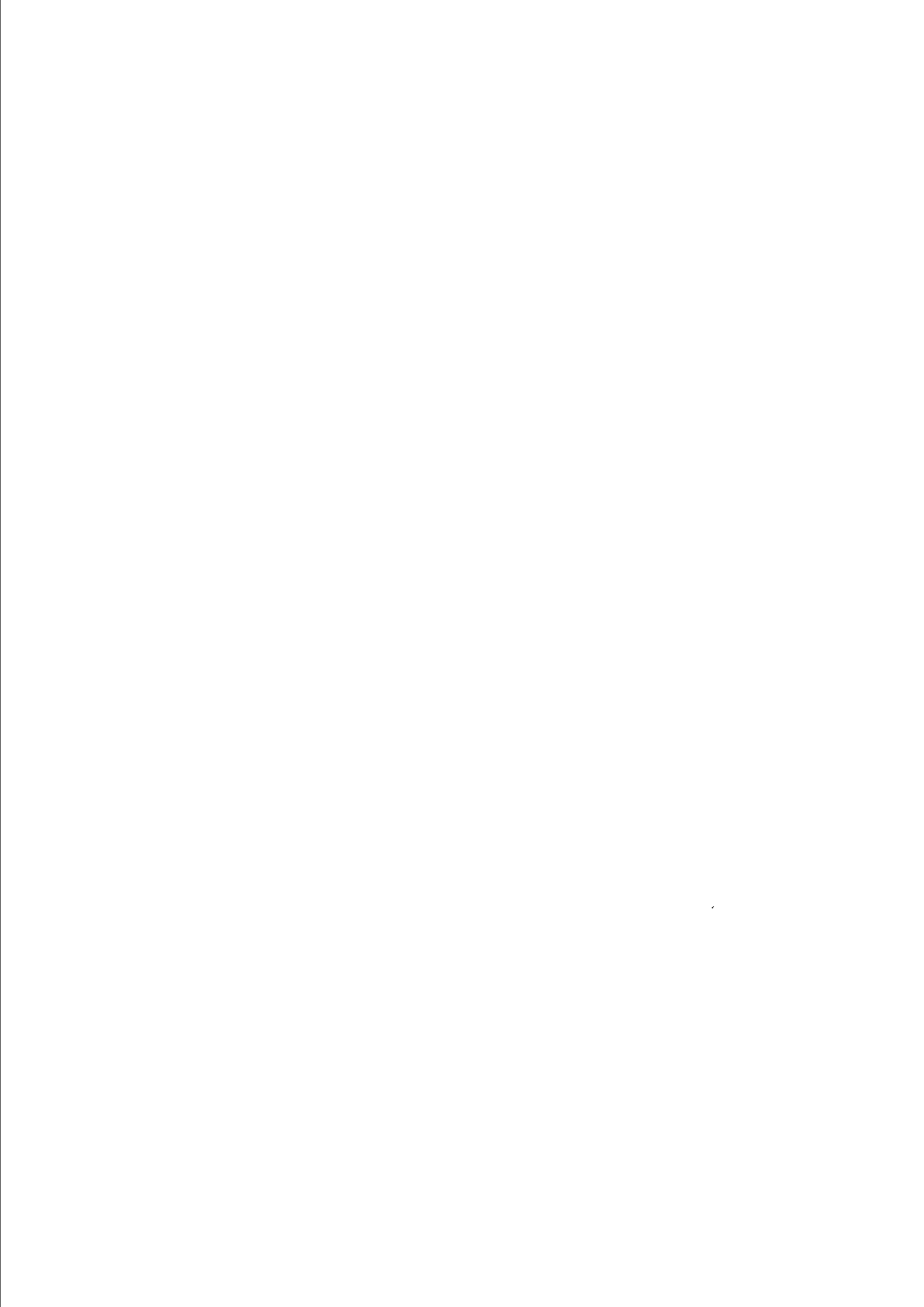


图 5－29 去掉网格线的曲面图

维绘图函数的使用，用户可以通过帮助系统进行学习，在此就不一一列举了。



第二篇

电路分析



第六章　电阻电路

电路理论中定义了一些理想元件：电压源、电流源、电阻、受控电压源、受控电流源、电容和电感。使用这些元件能够对实际电路建模，其模型在电路理论中简称为电路。由电阻、受控源和独立源组成的电路称为电阻电路，含有电容和电感等的电路称为动态电路。当电路模型给出后，在所有元件值和电源值给定的情况下，依据电路知识就可以写出有关数学方程。

MATLAB 具有强大的数学方程求解功能，读者在学习电路课时运用 MATLAB 可使注意力集中在电路分析方法本身上，而不会为数学方程的求解所困扰，从而有助于提高学习效率。

若用 MATLAB 只是求解电路方程，还不能完全减轻电路分析的工作量，因为电路方程的手工建立仍是一项十分艰巨的工作。借助 MATLAB 的编程语句，这一工作也可由计算机完成，从而在输入电路结构和元件值的情况下就能够分析电路。本章在介绍使用 MATLAB 求解电阻电路的基本方法之后，重点介绍作者用 MATLAB 语言编写的符号电路分析程序 sana，并给出一些分析实例。该程序具有非常简单的输入数据格式，适用于线性电路的分析。

6.1　电阻电路的方程及求解

手工建立电路方程的方法很多：列表法、支路法、网孔法、回路法和结点法。电阻电路的方程是代数方程，可表示为

$$\boldsymbol{A}\boldsymbol{x}=\boldsymbol{B}$$

其中 $\boldsymbol{x}$ 是由电路中的一些电压和电流构成的列向量；$\boldsymbol{A}$ 为系数矩阵，取决于电路元件的值；$\boldsymbol{B}$ 为右端列向量，其元素与电压源的电压和电流源的电流有关。

用 MATLAB 求解该线性代数方程的指令为

```
x = A\B
```

例 6－1　网孔方程的求解

电路如图 6－1 所示，已知 $R_1=6\ \Omega$，$R_2=4\ \Omega$，$R_3=3\ \Omega$，$v_{S1}=42\ \text{V}$，$v_{S2}=-10\ \text{V}$，

写出电路的网孔方程，并求电流 i_1 和 i_2。

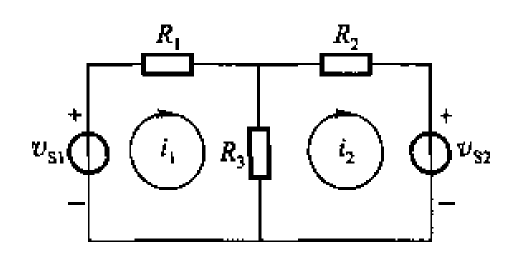


图 6－1　例 6－1 电路

解　对网孔 1 和网孔 2 分别应用 KVL 有

网孔 1：　$R_1 i_1 + R_3(i_1 - i_2) = v_{S1}$

网孔 2：　$R_2 i_2 + R_3(i_2 - i_1) = -v_{S2}$

整理以上方程后有

$$(R_1 + R_3)i_1 - R_3 i_2 = v_{S1}$$
$$-R_3 i_1 + (R_2 + R_3)i_2 = -v_{S2}$$

将其表示为矩阵

$$\begin{bmatrix} R_1 + R_3 & -R_3 \\ -R_3 & R_2 + R_3 \end{bmatrix}\begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} v_{S1} \\ -v_S \end{bmatrix}$$

求解上式矩阵的 M 文件如下：

```
r1 =6;r2 =4;r3 =3;vs1 =42;vs2 = -10;
a =[r1 +r3, -r3; -r3,r2 +r3]
b =[vs1; -vs2]
x =a\b
```

为方便起见，程序中采用小写英文字母，程序首先给出各元件值，其次建立有关系数矩阵，最后再求解方程。屏幕显示结果如下：

```
a =   9     -3
     -3      7
b =  42
     10
x =  6
     4
```

则网孔电流 $i_1 = 6$ A，$i_2 = 4$ A。

例 6－2　结点方程的求解

图 6－2 所示电路中，$R_1 = 1\ \Omega$，$R_2 = 2\ \Omega$，$R_3 = 3\ \Omega$，$R_4 = 4\ \Omega$，$i_S = 1$ A，电压控

制电流源的控制系数 $g=2$ S，写出电路的结点方程，并求出结点电压、电流 i_3 和独立电流源发出的功率。

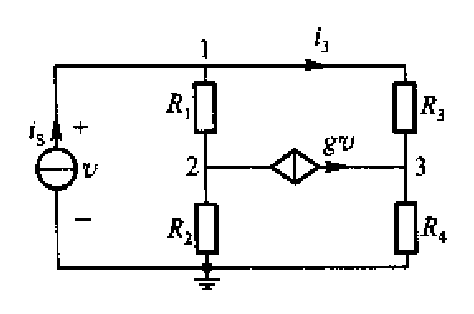


图 6-2　例 6-2 电路

解　设 $G_k=1/R_k(k=1,2,3,4)$，结点电压用 $v_k(k=1,2,3)$ 表示，把受控源的控制电压用方程变量表示，$v=v_1$，对各独立结点应用 KCL，有

结点 1：　$G_1(v_1-v_2)+G_3(v_1-v_3)=i_S$

结点 2：　$G_1(v_2-v_1)+G_2v_2+gv_1=0$

结点 3：　$G_3(v_3-v_1)+G_4v_3-gv_1=0$

整理以上三式，得

$$\begin{bmatrix} G_1+G_3 & -G_1 & -G_3 \\ g-G_1 & G_1+G_2 & 0 \\ -g-G_3 & 0 & G_3+G_4 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}=\begin{bmatrix} i_S \\ 0 \\ 0 \end{bmatrix}$$

用结点电压表示电流 i_3 和电流源发出的功率

$$i_3=G_3(v_1-v_3)$$

$$p=v_1i_S$$

M 文件如下：

```
g1 = 1/1;g2 = 1/2;g3 = 1/3;g4 = 1/4;
is = 1;g = 2;
a = [g1 + g3, - g1, - g3;g - g1,g1 + g2,0; - g - g3,0,g3 + g4]
b = [is;0;0]
v = a\b
i3 = g3 * (v(1) - v(3))
p = v(1) * is
```

屏幕显示为

```
a =      1.3333        -1        -0.33333
              1       1.5               0
```

```
  -2.3333         0            0.58333
b =   1
      0
      0
v =     1.5
       -1
        6
i3 =      -1.5
p =        1.5
```

即结点电压 $v_1=1.5$ V，$v_2=-1$ V，$v_3=6$ V，支路电流 $i_3=-1.5$ A，独立电流源发出的功率 $p=1.5$ W。

例 6-3 含有电流变量的结点法

图 6-3 所示电路含有电压源，已知 $v_A=10$ V，$R_1=2\ \Omega$，$R_2=5\ \Omega$，$R_3=2\ \Omega$，$v_B=2$ V，$r=3\ \Omega$，$R_4=4\ \Omega$，$i_C=1$ A，$g=-1$ S，用结点法求解电路的结点电压。

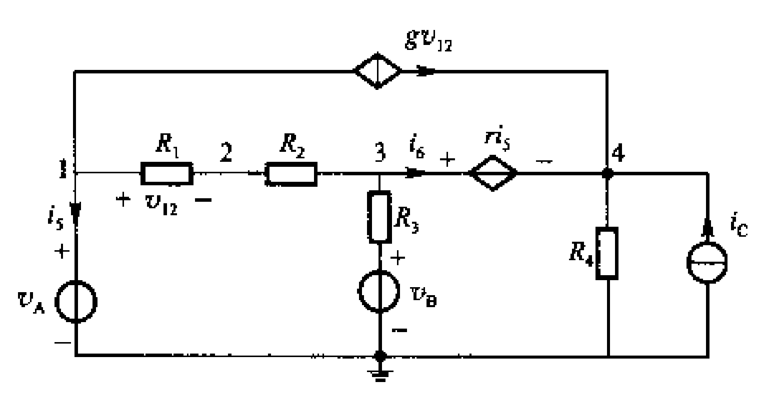


图 6-3 例 6-3 电路

解 当电路含有电压源支路时，为了能够对各结点应用 KCL 建立方程，并且能够根据方程变量的解答和支路 VCR（电压电流关系）求出其他电压和电流，在结点法中，除结点电压外，也把电压源的电流作为方程变量。图 6-3 所示电路中，有必要把电压源 v_A 的电流 i_5 和受控电压源的电流 i_6 作为方程变量，而电压源 v_B 由于有电阻与其串联，该支路的电流不必作为方程变量。因此，选取 v_1、v_2、v_3、v_4、i_5、i_6 为结点方程的变量，对各独立结点应用 KCL，可写出 4 个方程，其余两个方程由电流变量支路的 VCR 给出，可称为结点方程的补充方程。设 $G_k=1/R_k(k=1,2,3,4)$，结点方程为

结点 1：　$G_1(v_1-v_2)+g(v_1-v_2)+i_5=0$

结点 2：　$G_1(v_2-v_1)+G_2(v_2-v_3)=0$

结点 3: $G_2(v_3-v_2)+G_3(v_3-v_B)+i_6=0$

结点 4: $G_4v_4-g(v_1-v_2)-i_6=i_C$

i_5 支路: $v_1=v_A$

i_6 支路: $v_3-v_4-ri_5=0$

整理以上方程,并用矩阵形式表示,有

$$\begin{bmatrix} G_1+g & -G_1-g & 0 & 0 & 1 & 0 \\ -G_1 & G_1+G_2 & -G_3 & 0 & 0 & 0 \\ 0 & -G_2 & G_2+G_3 & 0 & 0 & 1 \\ -g & g & 0 & G_4 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -r & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ i_5 \\ i_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ G_3v_B \\ i_C \\ v_A \\ 0 \end{bmatrix}$$

M 文件如下:

```
g1 = 1/2;g2 = 1/5;g3 = 1/2;g4 = 1/4;
va = 10;vb = 2;ic = 1;
r = 3;g = -1;
a = zeros(6,6);
b = zeros(6,1);
a(1,1) = g1 + g;a(1,2) = -g1 - g;a(1,5) = 1;
a(2,1) = -g1;a(2,2) = g1 + g2;a(2,3) = -g2;
a(3,2) = -g2;a(3,3) = g2 + g3;a(3,6) = 1;b(3) = g3 * vb;
a(4,1) = -g;a(4,2) = g;a(4,4) = g4;a(4,6) = -1;b(4) = ic;
a(5,1) = 1;b(5) = va;
a(6,3) = 1;a(6,4) = -1;a(6,5) = -r;
x = a\b
```

由输出结果得

$v_1=10$ V, $v_2=7.8$ V, $v_3=2.3$ V, $v_4=-1$ V, $i_5=1.1$ A, $i_6=0.95$ A

该例题中,由于系数矩阵的零元素较多,先令其为零矩阵,然后对不为零的矩阵元素赋值。

以上三个例题显示,用 MATLAB 求解电路首先要依据电路知识手工建立电路方程,然后再编写有关程序,程序一般分为四部分:元件赋值,建立方程的系数矩阵,方程求解及输出量的求解。

6.2 建立结点方程的计算机方法

用手工方法建立方程的工作效率低下,而且易于出错,因此,该过程最好由计算机完成。在电路课程中,尽管学习了不少的方程建立方法,但适合计算机的并不多。一个比较好的方法应综合考虑以下几个因素:(1)容易编写建立电路方程的程序;(2)方程的变量数目不宜太多,否则需要采用求解方程的特殊方法,如稀疏矩阵技术;(3)方程建立方法对电路的结构不作特殊限制,适用于任意结构;(4)方程变量应尽可能是实际中最为关心的电路量,或根据方程变量能够容易求解其他电压和电流;(5)建立方程的运算量小。

网孔法只适合于平面电路,在计算机辅助分析中没有用处。回路法和割集法需要指定电路拓扑图的树,编程复杂,而且不满足条件(4)。支路法的变量数目较多,也要根据树才能确定独立回路,实际中很少采用。列表法的变量数目太多,只在极其特殊的一些大型软件中采用。与其他方法相比较,结点法在计算机辅助分析中具有明显的优势,对较为复杂一点的电路,结点方程的变量数目最少,而且适合任意结构的电路,对含有多端元件的电子电路,结点方程容易建立,也符合人们从多端元件角度考虑电路的习惯,此外,结点电压也是人们最为关心的电路量,根据结点电压和支路 VCR 能够很容易求得其他电压和电流。

结点方程的变量除结点电压外,还把一些电流作为变量。当把所有支路电流都作为变量时,其方程也称为列表方程。为了减少变量数目,对能根据支路 VCR 用方程变量表示其电流的支路,或电流已知的支路,如电阻、电流源,其电流就没有必要作为变量。因此,电流变量的选取应该满足在结点电压和这些电流变量已知的情况下,对电路中的任一条支路,能直接根据支路 VCR 用方程变量表示其电流。根据该要求,独立电压源、受控电压源的电流应该作为方程变量,此外,为方便起见,还常常把流控源的控制电流也作为方程变量。

结点方程由两种类型的方程组成:一类是对各独立结点应用 KCL 写出的方程,另一类是电流变量所在支路的电压电流关系式。注意,结点方程中不能含有方程变量以外的其余电压或电流,方程的数目应该等于方程变量的数目。

利用结点支路关联矩阵和支路 VCR,通过一定的矩阵运算可推导出结点方程,这种利用电路拓扑图建立结点方程的方法过程比较烦琐,其矩阵运算耗费不必要的机时,有关矩阵也占用一定的计算机存储空间。其实,仔细观察结点方程与电路之间的对应关系不难发现,每一种元件在方程中的贡献和元件值出现的位置与元件的类型和元件的连接结点有其对应关系。例如,图 6 - 2 所示电路

(重绘于图 6-4)的结点方程为

$$\begin{bmatrix} G_1+G_3 & -G_1 & -G_3 \\ g-G_1 & G_1+G_2 & 0 \\ -g-G_3 & 0 & G_3+G_4 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}=\begin{bmatrix} i_S \\ 0 \\ 0 \end{bmatrix}$$

电阻元件 G_1 与结点 1 和结点 2 相连接,其电导值 G_1 在方程系数矩阵中就出现在(1,1)、(2,2)的行和列位置,而电导 G_1 的负值出现在(1,2)、(2,1)位置,其他电阻元件也都有其对应关系。根据这一对应关系,将电路中所有元件在方程中的贡献全部完成后,方程也就建立起来了,这种建立结点方程的方法是一种填入法。

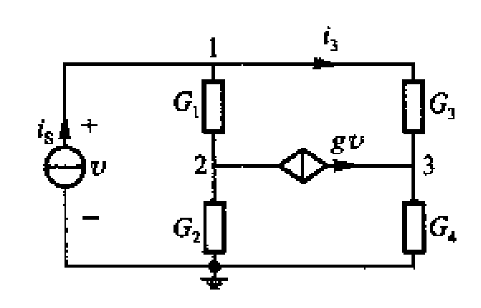


图 6-4 元件连接与结点方程的关系

手工列方程时,对简单电路往往能够直接写出,通过观察电路的连接关系,各结点的 KCL 方程可逐一给出。然而,结点方程的计算机建立方法一般按元件逐一填入有关矩阵,而不是按结点顺序,每处理一个元件,就将该元件的值填入方程中有关位置。

设电路是连通的,参考结点的编号规定为 0,其余结点采用正整数连续编号,方程变量采用先结点电压后电流的顺序排列,依据结点的 KCL 建立的方程,其顺序对应于结点编号,电流变量支路的 VCR 方程按电流变量的顺序排列。为了易于说明方程的填入方法,假设矩阵的行号和列号均从 0 开始,但要注意,MATLAB 中数组的标号只能大于 0,编程时需要作特殊处理。此外,假设每条支路只包含一个元件,即不采用复合支路的表示方式。下面讨论一些电路元件在方程中的填入规则。

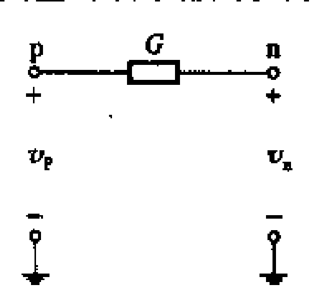


图 6-5 电阻元件

1. 电阻元件

参看图 6-5,设电阻 R 与结点 p 和结点 n 连接,结点 p 的电压为 v_p,结点 n 的电压为 v_n,电导 $G=1/R$,则流经该电阻的电流等于电导值乘以电阻上的电压。根据 KVL,任两个结点间的电压等于结点电压的差,于

是,结点 p 和结点 n 的 KCL 方程分别为

结点 p: $Gv_p - Gv_n + \cdots = 0$

结点 n: $-Gv_p + Gv_n + \cdots = 0$

式中的省略号表示与该结点相关联的其余支路的电流。由以上二式可得电阻元件在结点方程中的填入如图 6-6 所示。编程语句如下:

```
G = 1/R;
A(p,p) = A(p,p) + G;
A(p,n) = A(p,n) - G;
A(n,p) = A(n,p) - G;
A(n,n) = A(n,n) + G;
```

其中 A(p,p) = A(p,p) + G 表示原 A(p,p)的值加上 G 的值后再赋给 A(p,p),即

$$A(p,p) \Leftarrow A(p,p) + G$$

$$\boldsymbol{A} \Leftarrow \boldsymbol{A} + \begin{array}{c} \\ \\ p行 \\ \\ n行 \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccc} & p列 & & n列 & \end{array} \\ \left[\begin{array}{ccccc} & \vdots & & \vdots & \\ \cdots & G & \cdots & -G & \cdots \\ & \vdots & & \vdots & \\ \cdots & -G & \cdots & G & \cdots \\ & \vdots & & \vdots & \end{array}\right] \end{array}$$

图 6-6 电阻元件的填入

2. 电流源

参看图 6-7,电流源的电流 i_S 流出结点 p 并流入结点 n,KCL 方程为

结点 p: $\cdots = -i_S$

结点 n: $\cdots = i_S$

可见,电流源只对结点方程的右端向量有贡献,若在程序中用 is 表示电流源的电流,编程语句为

```
B(p) = B(p) - is;
B(n) = B(n) + is;
```

3. 电压源

参看图 6-8,设流经电压源的电流为 i_m,它流出结点 p 流入结点 n,i_m 为第

m 个方程变量，于是

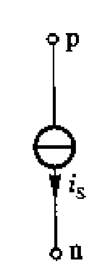


图 6－7 电流源

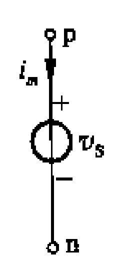


图 6－8 电压源

结点 p: $i_m + \cdots = 0$

结点 n: $-i_m + \cdots = 0$

支路: $v_p - v_n = v_S$

由于电流 i_m 为第 m 个方程变量，电压源支路的 VCR 方程在结点方程的第 m 行填入，编程语句如下：

```
A(p,m) = A(p,m) + 1;
A(n,m) = A(n,m) - 1;
A(m,p) = A(m,p) + 1;
A(m,n) = A(m,n) - 1;
B(m) = B(m) + vs;
```

4. 电压控制电流源

图 6－9 中，控制电压 v_c 为结点 p_c 与结点 n_c 之间的电压，即

$$v_c = v_{p_c} - v_{n_c}$$

按图中所示电压、电流方向，有

结点 p: $gv_{p_c} - gv_{n_c} + \cdots = 0$

结点 n: $-gv_{p_c} + gv_{n_c} + \cdots = 0$

由以上二式有

```
A(p,pc) = A(p,pc) + g;
A(p,nc) = A(p,nc) - g;
A(n,pc) = A(n,pc) - g;
A(n,nc) = A(n,nc) + g;
```

5. 电压控制电压源

参看图 6－10，将受控电压源支路的电流作为方程变量，设 i_m 为第 m 个方程变量，则结点 p 和结点 n 的 KCL 方程以及支路的 VCR 方程为

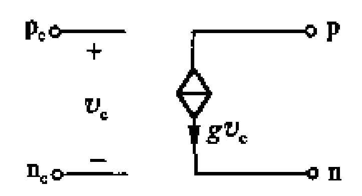


图 6-9 电压控制电流源

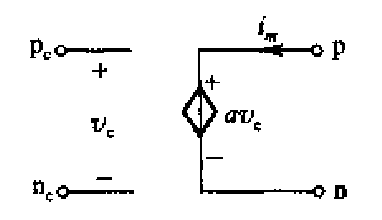


图 6-10 电压控制电压源

结点 p: $i_m + \cdots = 0$

结点 n: $-i_m + \cdots = 0$

支路: $v_p - v_n - a(v_{p_c} - v_{n_c}) = 0$

编程语句如下:

```
A(p,m) = A(p,m) + 1;
A(n,m) = A(n,m) - 1;
A(m,p) = A(m,p) + 1;
A(m,n) = A(m,n) - 1;
A(m,pc) = A(m,pc) - a;
A(m,nc) = A(m,nc) + a;
```

6. 电流控制电流源

为简化编程,不失一般性,假设控制电流只能是流经电压源的电流。在具体电路分析中,控制电流均可表示成这种形式,其方法是:在控制电流支路嵌入一个电压为零的电压源,显然,该电压源不会对电路产生任何影响。参看图 6-11,设控制电流 i_k 为第 k 个方程变量,则有

结点 p: $bi_k + \cdots = 0$

结点 n: $-bi_k + \cdots = 0$

编程语句如下:

```
A(p,k) = A(p,k) + b;
A(n,k) = A(n,k) - b;
```

注意,控制支路按电压源填入,此处不应重复填入。

7. 电流控制电压源

参看图 6-12,有关方程为

结点 p: $i_m + \cdots = 0$

结点 n: $-i_m + \cdots = 0$

支路: $v_p - v_n - ri_k = 0$

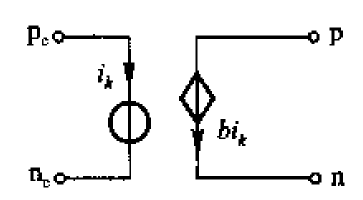


图 6－11 电流控制电流源

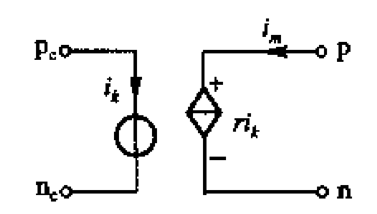


图 6－12 电流控制电压源

程序语句如下：

```
A(p,m) = A(p,m) + 1;
A(n,m) = A(n,m) - 1;
A(m,p) = A(m,p) + 1;
A(m,n) = A(m,n) - 1;
A(m,k) = A(m,k) - r;
```

8. 运算放大器

设图 6－13 所示运算放大器是理想的，如果将输出端的电流 i_m 也作为方程变量，有

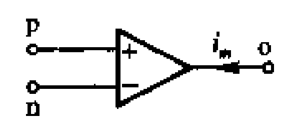


图 6－13 运算放大器

结点 o： $i_m + \cdots = 0$

电压关系： $v_p - v_n = 0$

运算放大器的电压关系式填入第 m 行，于是

```
A(o,m) = A(o,m) + 1;
A(m,p) = A(m,p) + 1;
A(m,n) = A(m,n) - 1;
```

如果视运算放大器的电压增益为有限值 a，则

结点 o： $i_m + \cdots = 0$

电压关系： $(v_p - v_n) - \frac{1}{a}v_o = 0$

这时，程序语句为：

```
A(o,m) = A(o,m) + 1;
A(m,p) = A(m,p) + 1;
A(m,n) = A(m,n) - 1;
A(m,o) = A(m,o) - 1/a;
```

以上讨论了电路元件在结点方程矩阵中的填入方法，在具体编写程序时，还有许多细节问题需要考虑：

（1）电路数据的输入。

电路数据包括各元件的名称、类型、连接结点和元件值。为了便于阅读，电路数据最好采用字符方式输入，为了便于记忆，元件的名称和类型最好用英文字母表示，如电阻的名称可以用以 R 打头的字符串表示，R 用以标识该元件为电阻。在编写大型软件时，也可编写绘制电路图的视窗软件，然后再将图形数据转化为对应的文本数据。

（2）变量排序。

为了程序使用方便起见，电路的原始结点编号可不连续或用字符串表示，但在程序中要对结点重新排序，建立结点编号对照表，此外要确定出结点总数。电流变量则要依据元件的类型确定，并对其排序。

（3）建立方程。

根据元件数据清单，对各个元件根据其类型逐一填写有关矩阵。在该过程执行之前，应该对这些矩阵赋以适当行数和列数的零矩阵。在所有元件填写完成后，还应删除矩阵的首行和首列，因为在填写矩阵时，为了减少判断结点是否为参考结点的语句，参考结点实质上也作为方程变量来填写有关矩阵，因此，在方程求解前必须对其删除。

（4）程序纠错。

用户在使用程序时，难免会出现不符合输入数据格式要求的输入形式，一个好的程序应对这种错误给出必要的警告或中断程序继续执行的错误信息提示，待输入错误完全纠正后程序再执行后续过程。

（5）输出数据。

程序的输出不仅能够给出方程变量的解答，而且可以根据用户要求给出其余电路量的计算结果。有时，有必要给出电路方程的系数矩阵，以便用户检查输入数据的正确性。

下面用图 6－4 说明结点方程的填入过程。设元件数据清单中元件的顺序依次为 R_1、R_2、R_3、R_4、独立电流源和受控电流源。由于共有 4 个结点，且没有电流变量，先对矩阵 $\boldsymbol{A}$ 赋以 4×4 的零矩阵，对矩阵 $\boldsymbol{B}$ 赋以 4×1 的零矩阵，各元件填入后的 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 依次为

填入 R_1

$$\boldsymbol{A}=\begin{bmatrix}0 & 0 & 0 & 0\\ 0 & G_1 & -G_1 & 0\\ 0 & -G_1 & G_1 & 0\\ 0 & 0 & 0 & 0\end{bmatrix} \qquad \boldsymbol{B}=\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$$

填入 R_2 $A=\begin{bmatrix} G_2 & 0 & -G_2 & 0 \\ 0 & G_1 & -G_1 & 0 \\ -G_2 & -G_1 & G_1+G_2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ $B=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

填入 R_3 $A=\begin{bmatrix} G_2 & 0 & -G_2 & 0 \\ 0 & G_1+G_3 & -G_1 & -G_3 \\ -G_2 & -G_1 & G_1+G_2 & 0 \\ 0 & -G_3 & 0 & G_3 \end{bmatrix}$ $B=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

填入 R_4 $A=\begin{bmatrix} G_2+G_4 & 0 & -G_2 & -G_4 \\ 0 & G_1+G_3 & -G_1 & -G_3 \\ -G_2 & -G_1 & G_1+G_2 & 0 \\ -G_4 & -G_3 & 0 & G_3+G_4 \end{bmatrix}$ $B=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

填入独立电流源 $A=\begin{bmatrix} G_2+G_4 & 0 & -G_2 & -G_4 \\ 0 & G_1+G_3 & -G_1 & -G_3 \\ -G_2 & -G_1 & G_1+G_2 & 0 \\ -G_4 & -G_3 & 0 & G_3+G_4 \end{bmatrix}$ $B=\begin{bmatrix} -i_S \\ i_S \\ 0 \\ 0 \end{bmatrix}$

填入受控电流源 $A=\begin{bmatrix} G_2+G_4 & 0 & -G_2 & -G_4 \\ 0 & G_1+G_3 & -G_1 & -G_3 \\ -g-G_2 & g-G_1 & G_1+G_2 & 0 \\ g-G_4 & -g-G_3 & 0 & G_3+G_4 \end{bmatrix}$ $B=\begin{bmatrix} -i_S \\ i_S \\ 0 \\ 0 \end{bmatrix}$

删除首行和首列 $A=\begin{bmatrix} G_1+G_3 & -G_1 & -G_3 \\ g-G_1 & G_1+G_2 & 0 \\ -g-G_3 & 0 & G_3+G_4 \end{bmatrix}$ $B=\begin{bmatrix} i_S \\ 0 \\ 0 \end{bmatrix}$

这两个矩阵则为结点方程的系数矩阵。

6.3 符号电路分析程序 sana

用符号表示元件值求解电路响应的过程称为符号电路分析。与数值方法相比较,符号分析具有以下优点:(1)电路响应的符号表达式给出了响应与元件值之间的关系,在电路设计中根据这一关系可以确定元件值。(2)符号分析的计算精度较高,有时能够得到绝对精确的计算结果。(3)根据表达式计算响应要

比电路方程节省计算时间。

在手工电路分析中,为了便于公式推导,一般采用器件的理想模型。尽管如此,当电路比较复杂时,公式推导过程仍将变成一项非常繁重的工作。另一方面,为了能够设计出性能卓越的电路,还应尽可能地采用能够反映电路真实行为的器件模型。随着模拟集成电路工作频率的提高,分析器件的寄生参数对电路性能的影响变得十分重要。然而,当器件的模型比较复杂时,手工推导电路响应的表达式具有相当大的难度。

MATLAB 软件的符号工具箱提供了求导数、积分、极限、代数方程、微分方程等的数学工具,利用它也可对电路问题进行分析。MATLAB 中求解符号代数方程组

$$\boldsymbol{Ax} = \boldsymbol{B}$$

的指令为

```
x = A\B
```

该指令使用高斯消去法求解 x 的符号解。

本节介绍作者用 MATLAB 语言编写的符号电路分析函数 sana.m,该程序用结点法建立电路方程,以结点电压和电压源、受控电压源的电流为变量,流控源的控制电流限制为电压源和受控电压源的电流,建立方程的原理如上节所述。结点编号必须为整数,可不连续,参考结点的编号必须为数字 0。

程序 sana 的调用格式为

```
sana(ckt);
[x,B,A] = sana(ckt)
```

其中,输入变元 ckt 为描述电路的字符群数组(cell array),每一数组描述一个元件;x 为方程变量的列向量;B 为方程的右端向量;A 为方程的系数矩阵。

元件的描述格式一般为

'元件名称 连接结点 元件值'

其中,元件名称不区分英文字母的大小写,所有电阻的名称可相同,但电压源等以电流为变量的元件的名称不得一样,元件的类型用元件名称的首字母区分,如电阻元件用以字母 R 或 r 打头的字符串表示;元件值既可用字符串表示,也可用具体的数值给出,但要注意,大写字母元件值与对应的小写字母有不相同的含义。元件值字符串中也可包含 MATLAB 中规定的符号常数和函数,也可以是一个表达式。

(1) 电阻元件。

格式为:'r* * * 正端 负端 电阻值'

程序中,电压和电流的方向均规定从元件的正端结点指向负端结点,电阻值不能为零。如

'ra 3 5 R'

表示电阻 ra 与结点 3 和结点 5 连接,电阻值用 R 表示。

(2) 电流源。

格式为:'i* * * 正端 负端 电流值'

电流的方向规定从元件的正端结点指向负端结点。

(3) 电压源。

格式为:'v* * * 正端 负端 电压值'

正端结点为电压的正极性端,电压源的电流从正端指向负端。

(4) 电压控制电流源。

格式为:'g* * * 正端 负端 正控制端 负控制端 控制系数'

(5) 电压控制电压源。

格式为:'e* * * 正端 负端 正控制端 负控制端 控制系数'

(6) 电流控制电流源。

格式为:'g* * * 正端 负端 控制支路的元件名称 控制系数'

注意,控制电流只能是独立电压源、受控电压源和运算放大器输出端的电流,后者电流指向运算放大器。

(7) 电流控制电压源。

格式为:'e* * * 正端 负端 控制支路的元件名称 控制系数'

(8) 运算放大器。

格式为:'a* * * 同相端 反相端 输出端 增益'

增益值缺省时,指增益为无限大,这时表示理想运算放大器。

有关元件格式的说明可在 MATLAB 的工作区使用

help sana

获得帮助。

例如,分析图 6-4 所示电路的 M 文件如下:

```
ckt = {'r1  1  2  R1'
       'r2  2  0  R2'
       'r3  1  3  R3'
       'r4  3  0  R4'
       'i   0  3  is'
       'g1  2  3  1  0  g'};
```

```
[x,b,a] = sana(ckt)
```

其中,描述电路的群数组为 ckt,群数组使用花括号,语句[x,b,a] = sana(ckt)打印出结点方程的矩阵。输出结果如下:

```
Nodal voltages v * * and element currents i * * :
v1 = -1. * is * R4 * (R2 + R1)/( -1. * R3 -1. * R3 * R2 * g + R1 * R4 *
g -1. * R2 -1. * R4 -1. * R1)
v2 = R4 * is * R2 * ( -1. + g * R1)/( -1. * R3 -1. * R3 * R2 * g + R1 *
R4 * g -1. * R2 -1. * R4 -1. * R1)
v3 = -(R2 + R1 + R3 * R2 * g + R3) * is * R4/( -1. * R3 -1. * R3 * R2 *
g + R1 * R4 * g -1. * R2 -1. * R4 -1. * R1)
x =
[ -is * R4 * (R2 + R1)/( -R3 - R3 * R2 * g + R1 * R4 * g - R2 - R4 -
R1)]
[R4 * is * R2 * ( -1 + g * R1)/( -R3 - R3 * R2 * g + R1 * R4 * g - R2 - R4 -
R1)]
[ -(R2 + R1 + R3 * R2 * g + R3) * is * R4/( -R3 - R3 * R2 * g + R1 * R4 *
g - R2 - R4 - R1)]
b =
[  0]
[  0]
[  is]
a =
[  1/R1 + 1/R3,        -1/R1,         -1/R3]
[     -1/R1 + g,  1/R1 + 1/R2,             0]
[     -1/R3 - g,            0,   1/R3 + 1/R4]
```

结点电压用字母 v 后紧跟结点号的变量表示,电流用以字母 i 后紧跟元件名的变量表示,它们在 MATLAB 工作区作为变量还可参与其他一些运算,例如:

```
v = v1 - v2
```

表示结点 1 与结点 2 间的电压。

有时,符号输出结果的分子和分母不易辨认,可用指令

```
pretty(v1)
```

显示结点 1 的电压。显示结果为

```
                 is   R4   (R2 + R1)
   -    -----------------------------------------------
         - R3 - R3 R2 g + R1 R4 g - R2 - R4 - R1
```

例 6-4 电阻电路分析

电路如图 6-14 所示，已知 $i_S = 2.5$ A，$R_1 = 2\ \Omega$，$R_2 = 0.4\ \Omega$，$R_3 = 1\ \Omega$，$R_4 = 3\ \Omega$，$v_a = 0.5\ v_1$，$i_b = 1 \times v_1$，求结点 3 与结点 2 之间的电压。

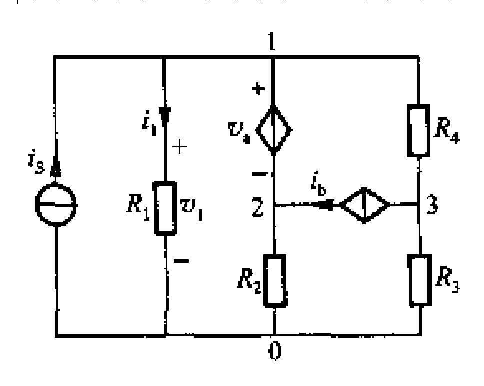


图 6-14 例 6-4 电路

解 M 文件如下：

```
ckt = {'is 0 1 2.5'
       'r1 1 0 2'
       'e 1 2 1 0 0.5'
       'r4 1 3 3'
       'g 3 2 1 0 1'
       'r2 2 0 0.4'
       'r3 3 0 1'};
sana(ckt);
v32 = v3 - v2
```

输出结果为

```
v1 = 2.
v2 = 1.
v3 = -1.
ie = .5000
v32 = -2.
```

则结点 3 与结点 2 之间的电压为 -2 V。

如果元件值以分式形式表示，程序执行后的输出结果也将以分式形式给出。把分式转化为小数形式的函数为

numeric

6.4 电阻电路分析举例①

本节针对电阻电路的主要知识点,用一些例题说明 sana 在电路分析中的应用。对正在学习电路课程的读者,用该程序可对相当多的题目进行辅助分析。

例 6-5 等效电阻

图 6-15 所示电路中,设各电阻的电阻值均为 R,求结点 1 与结点 0 之间的等效电阻。

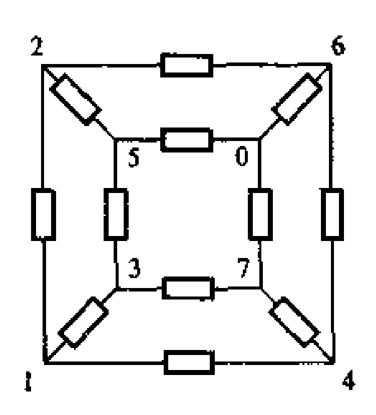


图 6-15 例 6-5 电路

解 仅由电阻元件构成的单口电路,其等效电阻也可用结点法求解,而不必使用星形/三角形变换方法。对复杂电路,用计算机求解非常方便。若给端口施加一个电流源 i,用结点法求出端口电压 v,若 v 和 i 对端口取关联方向,则等效电阻为

$$R_{eq}=\frac{v}{i}$$

设取 $i=1$ A,求等效电阻的 M 文件如下:

```
ckt = {'r 1 2 R'
       'r 1 3 R'
       'r 1 4 R'
       'r 5 0 R'
       'r 6 0 R'
       'r 7 0 R'
```

① 本节的一些例题取自《电路》(第四版)(邱关源主编.北京:高等教育出版社,1999)中的习题。

```
        'r 2 5 R'
        'r 2 6 R'
        'r 3 5 R'
        'r 3 7 R'
        'r 4 6 R'
        'r 4 7 R'
        'i 0 1 1'};
sana(ckt)
req = v1
```

执行程序后得

$$v_2 = v_3 = v_4 = \frac{1}{2}R$$

$$v_5 = v_6 = v_7 = \frac{1}{3}R$$

$$R_{eq} = \frac{5}{6}R$$

可见,结点 2、3、4 的电压相等,结点 5、6、7 的电压也相等,这是由电路的结构所决定的。

例 6-6　元件值的求解

图 6-16 所示电路为由桥 T 电路构成的电压衰减器,若 $R_{ab} = R_L$,求电阻 R_2 应该满足的关系式。

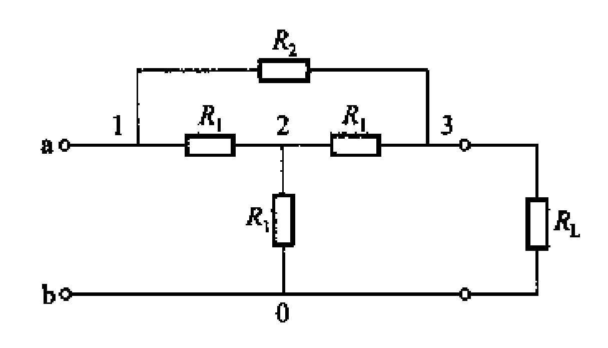


图 6-16　例 6-6 电路

解　给端口 ab 施加一个 1 A 的电流源,求出结点 1 的电压,该电压在数值上等于端口 ab 的等效电阻 R_{ab}。用符号代数方程求解函数 solve 求解方程 $R_{ab} - R_L = 0$,即可得到电阻 R_2 的值。

```
ckt = {'r 1 2 R1'
```

```
        'r 2 0 R1'
        'r 2 3 R1'
        'r 1 3 R2'
        'r 3 0 RL'
        'i 0 1 i'};
sana(ckt);
eq = v1 - sym('RL');
R2 = solve(eq,'R2')
```

执行程序后得

$$R_2 = \frac{2R_1R_L^2}{3R_1^2 - R_L^2}$$

例 6-7 结点方程

图 6-17 所示电路含有电压源，建立该电路的结点方程并求解。

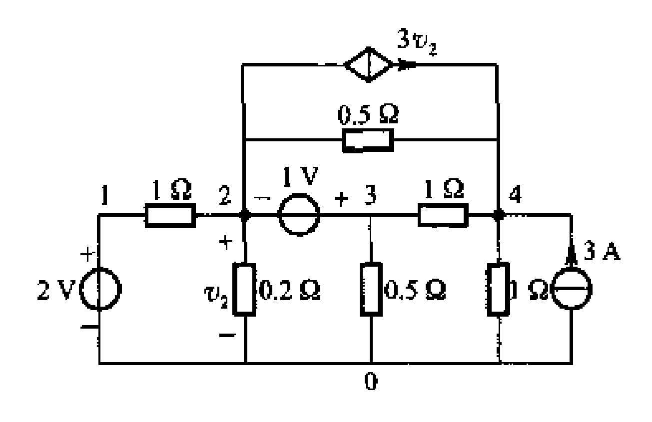


图 6-17 例 6-7 电路

解 手工列方程时，电阻与电压源的串联可作为一条支路处理，但用程序求解时按两条支路对待。如果元件值以分数形式输入，M 文件如下：

```
ckt = {'r 1 2 1'
       'v1 1 0 2'
       'r 2 0 1/5'
       'v2 3 2 1'
       'r 2 4 1/2'
       'g 2 4 2 0 3'
       'r 3 0 1/2'
       'r 3 4 1'
       'r 4 0 1'
       'i 0 4 3'};
```

```
[x,b,a] = sana(ckt)
```

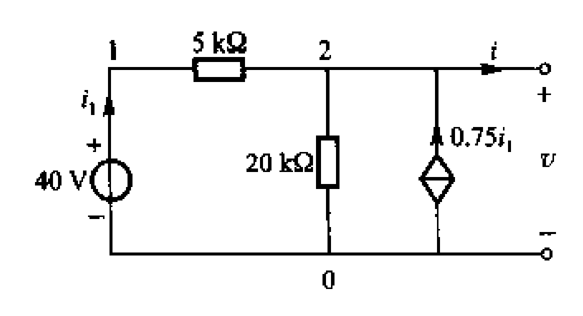
打印结果为

```
v1 =2.
v2 = .2105
v3 =1.211
v4 =1.316
iv1 = -1.789
iv2 = -2.316
x =
[         2]
[      4/19]
[     23/19]
[     25/19]
[    -34/19]
[    -44/19]
b =
[ 0]
[ 0]
[ 0]
[ 3]
[ 2]
[ 1]
a =
[  1,  -1,   0,   0,  1,   0]
[ -1,  11,   0,  -2,  0,  -1]
[  0,   0,   3,  -1,  0,   1]
[  0,  -5,  -1,   4,  0,   0]
[  1,   0,   0,   0,  0,   0]
[  0,  -1,   1,   0,  0,   0]
```

注意,电压源电流的方向从正端指向负端,输出变量的顺序如打印结果所示。若在程序中的 sana 函数之后不加分号“;”,方程变量的解既给出了小数形式,也给出了分数形式。

例 6-8　戴维宁等效电路

求图 6-18 所示电路的戴维宁等效电路。



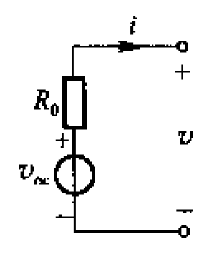


图 6-18　例 6-8 电路

解　若给端口施加一个符号电流源 i,求出端口电压 v,则 v 可表示为

$$v = v_{oc} - R_0 i$$

于是 v_{oc} 为戴维宁电压,R_0 为戴维宁电阻。M 文件如下:

```
ckt = {'v 1 0 40'
       'r 1 2 5e3'
       'r 2 0 20e3'
       'g 0 2 v -0.75'
       'i 2 0 i2'};
sana(ckt);
v2
```

注意:由于程序规定电压源的电流从正端指向负端,因此,受控电流源的控制系数在输入时应按 -0.75 输入。此外,电流源的电流值不能用字母 i 表示,因为 MATLAB 中 i 为复数的虚单位。程序求出

```
v2 = 35. -2500. * i2
```

则

$$v_{oc} = 35\ \text{V}, \quad R_0 = 2\ 500\ \Omega$$

例 6-9　含有运算放大器的电路

图 6-19 所示电路为差分放大器电路,求下述两种情况下的输出电压 v_o。(1)视运算放大器是理想的;(2)设运算放大器的电压增益为有限值 a。

解　(1) M 文件如下:

```
ckt = {'V1 1 0 V1'
       'V2 2 0 V2'
       'R 1 3 R1'
```

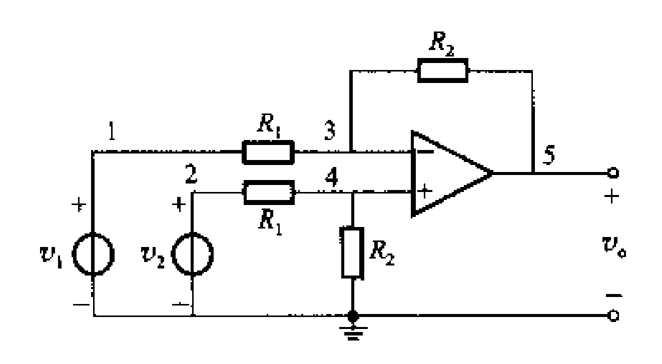


图 6-19 差分放大器电路

```
        'R 2 4 R1'
        'R 4 0 R2'
        'R 3 5 R2'
        'A 4 3 5'};
    sana(ckt);
    vo = simple(v5)
```

其中,simple 函数对表达式 v5 进行化简。

输出结果为

```
v1 = V1
v2 = V2
v3 = V2 * R2/(R2 + R1)
v4 = V2 * R2/(R2 + R1)
v5 = R2/R1 * V2 - 1. * R2/R1 * V1
iv1 = -1./R1 * V1 + 1/R1/(R2 + R1) * V2 * R2
iv2 = -1. * V2/(R2 + R1)
ia = 1/R1 * V1 - 1./R1/(R2 + R1) * V2 * R2
vo = -( - V2 + V1) * R2/R1
```

则输出电压为

$$v_o = \frac{R_2}{R_1}(v_2 - v_1)$$

该电路具有差分放大功能。

(2) 当运算放大器的增益为有限值时,运算放大器可用电压控制电压源描述,或将第(1)问中的有关语句修改为

```
'A 4 3 5 a'
```

执行程序后得

$$v_o = \frac{aR_2}{R_2 + R_1 + aR_1}(v_2 - v_1)$$

例 6－10 双口电路的电压控制型参数和方程

参看图 6－20，当双口电路含有独立电源时，电压控制型方程为

$$i_1 = g_{11}v_1 + g_{12}v_2 + i_{S1}$$

$$i_2 = g_{21}v_1 + g_{22}v_2 + i_{S2}$$

为了求得方程中的各个参数，可给二个端口分别施加电压源 v_1 和 v_2，然后再求解端口电流 i_1 和 i_2，根据 i_1 和 i_2 的表达式可确定出各个参数。对图 6－20 所示双口电路，求电压控制型方程和参数。

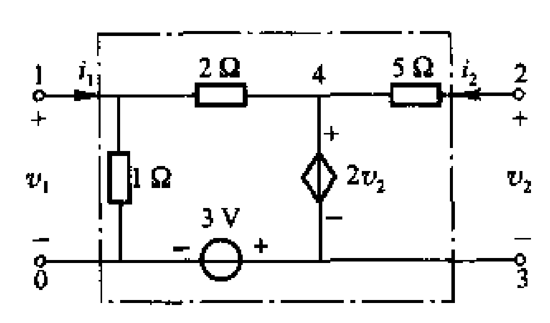


图 6－20 例 6－10 电路

解 M 文件如下：

```
ckt = {'r 1 0 1'
       'r 1 4 2'
       'r 4 2 5'
       'v 3 0 3'
       'e 4 3 2 3 2'
       'v1 1 0 v1'
       'v2 2 3 v2'};
sana(ckt);
i1 = -iv1
i2 = -iv2
```

计算结果为

```
i1 = 3/2 * v1 - v2 - 3/2
i2 = -1/5 * v2
```

则

$$g_{11}=\frac{3}{2},\quad g_{12}=-1,\quad i_{S1}=-\frac{3}{2}$$

$$g_{21}=0,\quad g_{22}=-\frac{1}{5},\quad i_{S2}=0$$

例 6-11 双口电路的混合参数和方程

双口电路的混合参数方程为

$$v_1=h_{11}i_1+h_{12}v_2+v_S$$
$$i_2=h_{21}i_1+h_{22}v_2+i_S$$

求图 6-20 所示双口电路的混合参数。

解 给端口 1 加电流源,给端口 2 加电压源,求出端口 1 的电压和端口 2 的电流,则可得到混合参数。M 文件如下:

```
ckt = {'r 1 0 1'
       'r 1 4 2'
       'r 4 2 5'
       'v 3 0 3'
       'e 4 3 2 3 2'
       'i1 0 1 i1'
       'v2 2 3 v2'};
sana(ckt);
v1
i2 = -iv2
```

计算结果为

```
v1  =  1 +2/3 * i1 +2/3 * v2
i2  =  -1/5 * v2
```

即

$$h_{11}=\frac{2}{3}\ \Omega,\quad h_{12}=\frac{2}{3},\quad v_S=1\ \text{V}$$

$$h_{21}=0,\quad h_{22}=-\frac{1}{5}\ \text{S},\quad i_S=0\ \text{A}$$

第七章　动态电路的时域分析

由电容、电阻、受控源和独立源组成的电路，若电容能够用电压源替代，则替代后的电路相当于电阻电路，流经电容的电流可表示为电容电压的函数。又根据电容元件的 VCR，电容值乘以电容电压的导数为流经该电容的电流，因此，对每一电容元件，其电压的导数就为各电容电压的函数。

当电路既含有电容又含有电感时，若电容能用电压源替代，电感能用电流源替代，则电容电压的导数为电容电压和电感电流的函数，电感电流的导数也为电容电压和电感电流的函数。当对电路中的所有电容和电感写出上述方程后，这一组方程就为电路的微分方程，方程的数目等于动态元件的数目。

当电路的微分方程建立后，微分方程的求解就可用 MATLAB 指令实现。在 MATLAB 中，微分方程的符号求解指令为 dsolve，数值求解指令根据所采用的算法不同共有 7 种，它们具有相同的调用格式。

从微分方程求得电容电压和电感电流后，把电容用电压源替代，电感用电流源替代，运用电阻电路的分析方法就可求出其他电压和电流。

尽管目前市场上已有许多性能良好的电路分析软件，但由于微分方程的求解至今还没有一个适合各种情况的高效算法，因而电路通用软件中所采用的算法也就不可能适应于各种电路的分析。与现有的一些电路分析软件相比，使用 MATLAB 分析电路有一定的优势：(1) MATLAB 是一种功能强大的数学软件，适合于各个学科，它在本科生中的普及越来越广，而专用于电路分析的工程软件在手头并不一定具备。(2) MATLAB 不仅提供了多种数值积分方法，它还具备强大的矩阵运算、程序编写、图形绘制等许多功能，它的工具箱也将越来越丰富，在此基础上，用户便于开发自己的应用程序。(3) 对从事电路科研工作的人员，使用 MATLAB 可以解决一些现有电路分析软件不能解决的问题。

对复杂电子电路，要建立电路的微分方程并不是一项简单的工作，至今还没有一个令人十分满意的方法。因此，在大型电子电路分析软件中，一般不采用这种方法，而是直接将某些数值积分方法用于电容和电感元件，得到离散时间电路模型，然后再用结点法进行分析。本章对这些内容也将给予简要介绍，主要是为

了让读者对这种方法有一定的了解，在此基础上可以编写自己的电路分析程序，相应的内容有：求解常微分方程的后向欧拉法和 Gear 法，离散时间电路模型的概念以及离散时间结点方程的建立方法。

当电路的微分方程是线性方程时，也可采用 MATLAB 控制系统工具箱提供的指令求方程的数值解，这时，微分方程也可以是单变量的输入输出方程，有关内容将在第三篇第十章中介绍。线性电路的动态过程也可采用单边拉普拉斯变换法分析，详见第三篇第十一章内容。

7.1 一阶电路

当电路的动态元件只有一个时，电路其余部分在动态元件处可构成单口电阻电路。设动态元件为电容，则电路可表示为图 7－1(a)所示的连接形式，如果把单口电阻电路 N_R 用最简等效电路表示，设等效电流源的电流为 $i(t)$，等效电阻为 R，则电路如图 7－1(b)所示。

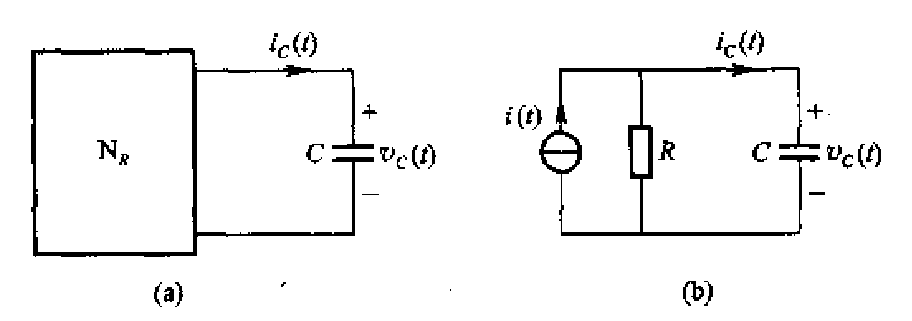


图 7－1 由电容组成的一阶电路

以电容电压 $v_C(t)$ 为变量的微分方程为

$$C\frac{\mathrm{d}v_C(t)}{\mathrm{d}t}+\frac{1}{R}v_C(t)=i(t) \tag{7-1}$$

设电容电压的 0_+ 初始值 $v_C(0)$ 给定，当 $i(t)$ 为直流电流 $i(t)=I$ 时，可解出

$$v_C(t)=RI+[v_C(0)-RI]\mathrm{e}^{-\frac{1}{RC}t} \tag{7-2}$$

当 $i(t)$ 为时变电流时，$v_C(t)$ 则为微分方程的齐次解与特解之和，可表示为

$$v_C(t)=A\mathrm{e}^{-\frac{1}{RC}t}+v_{Cp}(t) \tag{7-3}$$

$v_{Cp}(t)$ 根据微分方程右端的函数形式确定，令式(7－3)满足初始值即可确定出常数 A，最终得：

$$v_C(t)=[v_C(0)-v_{Cp}(0)]\mathrm{e}^{-\frac{1}{RC}t}+v_{Cp}(t) \tag{7-4}$$

求解图 7－1(a)所示电路的关键是列出以电容电压为变量的微分方程，这一过程能够用电路分析函数 sana 实现。把电容用电压源 v_C 替代，用 sana 求出流经电容的电流 i_C，其中，i_C 的表达式是 v_C 的函数，设 $i_C = f(t, v_C)$，则以电容电压为变量的微分方程可表示为

$$C\frac{\mathrm{d}v_C(t)}{\mathrm{d}t} - f(t, v_C) = 0 \tag{7-5}$$

求解微分方程可用 MATLAB 符号工具箱提供的函数 dsolve 完成，其调用格式为

```
[x1,x2,…] = dsolve (eqn1,eqn2,…)
[x1,x2,…] = dsolve (eqn1,eqn2, …,'t')
```

其中，函数的输入变元为表示微分方程及其初始值的字符串变量，可以有多个，与方程的阶次和初始条件的数目有关，各输入变元以逗点分开。在每一方程中，各变量的隐含自变量是时间 t。在字符串中，一阶求导运算$\frac{\mathrm{d}}{\mathrm{d}t}$用 D 表示，二阶求导运算$\frac{\mathrm{d}^2}{\mathrm{d}t^2}$用 D2 表示，其余类推。注意，变量名不能以字母 D 打头。输入变元 t 表示自变量，可省去。输出变元 x1、x2 等为方程变量的解答。

例如，求解微分方程

$$\begin{cases}\dfrac{\mathrm{d}y}{\mathrm{d}t} = -ay \\ y(0) = 1\end{cases}$$

的语句为

```
dsolve('Dy = -a*y','y(0) = 1')
```

用函数 dsolve 求出电容电压后，把电容用电压源替代，如图 7－2 所示，该电路为电阻电路，用它可求出其他电压和电流。

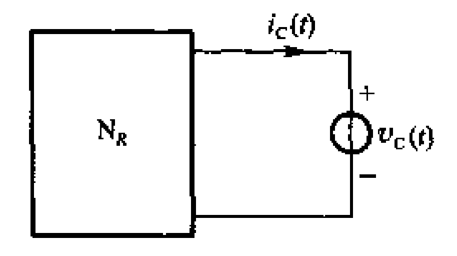


图 7－2 输出量的求解

根据以上所述，计算一阶电容电路的过程为：(1)用符号表示电容电压，把电容用电压源替代，编写描述电路的群数组，用函数 sana 求出电容电流；(2)构

造以电容电压为变量的微分方程和初始值的字符串变量，用函数 dsolve 求解电容电压；(3)根据 sana 函数的解答计算其他电压和电流。

例 7-1 一阶电容电路

图 7-3 所示电路中，$R_1=1\ \Omega$，$R_2=2\ \Omega$，$v_S=2$ V，$g=0.5$ S，$C=0.5$ F，电容电压的初始值 $v_C(0)=1$ V，求电容电压 v_C 和电压 v_A。

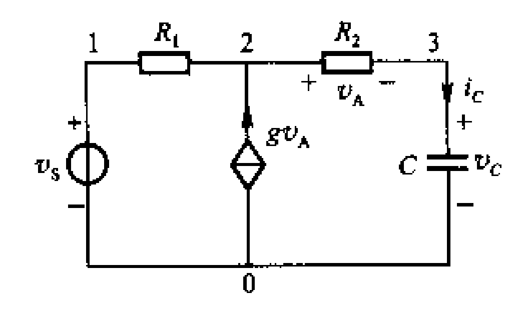


图 7-3 例 7-1 电路

解 电容电压用 vc 表示，M 文件如下：

```
ckt = {'vc 3 0 vc'
       'r 1 2 1'
       'r 2 3 2'
       'v 1 0 2'
       'g 0 2 2 3 0.5'};
sana(ckt);
eq1 = char('0.5 * Dvc' - ivc);                    % ---(1)
vc = dsolve(eq1,'vc(0) = 1')                      % ---(2)
va = v2 - v3                                      % ---(3)
va = subs(va,'vc',vc)                             % ---(4)
```

其中，语句(1)将 $C\frac{\mathrm{d}v_C}{\mathrm{d}t}-i_C$ 的符号表达式转化为字符串，因电容用电压源 vc 替代，则该电压源的电流 ivc 为原电路中流经电容的电流；语句(2)用函数 dsolve 求解电容初始电压值 $v_C(0)=1$ V 的微分方程；语句(3)求电压 v_A；语句(4)用求出的电容电压替代表达式中的 v_C。屏幕显示为

```
v3 = vc
v1 = 2.
v2 = 2.
ivc = -.5000 * vc + 1.
```

```
iv =0
vc = 2 - exp( - t)
va = 2. - vc
va = exp( - t)
```

即

$$v_C = 2 - e^{-t}\ \mathrm{V}$$

$$v_A = e^{-t}\ \mathrm{V}$$

例 7-2 输入信号为指数函数的一阶电容电路

上例中若 $v_S = 2e^{-t}$ V，电容电压的初始值 $v_C(0) = 1$ V，求电容电压 v_C 和电压 v_A。

解 将上例中描述电压源的字符数组修改为

```
'v 1 0 2 * exp( - t)'
```

可求得

$$v_C = (2t + 1)e^{-t}\ \mathrm{V}$$

$$v_A = (-2t + 1)e^{-t}\ \mathrm{V}$$

若一阶电路的动态元件为电感，电路如图 7-4(a)所示。建立微分方程的步骤为：用符号表示电感电流，把电感用电流源替代，见图 7-4(b)，编写描述该电路的群数组，用函数 sana 求出电感电压的表达式，它是输入信号和电感电流的函数，则微分方程为

$$L\frac{di_L}{dt} - f(t, i_L) = 0$$

根据上式建立表示微分方程和初始值的字符串变量，用函数 dsolve 求解电感电流。最后，利用电感电流的解答计算其他电压和电流。

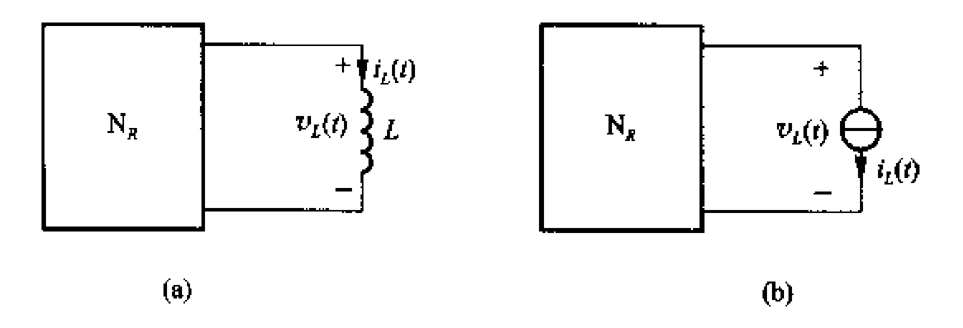


图 7-4 由电感组成的一阶电路

例 7-3 一阶电感电路

图 7-5 所示电路在 $t = 0$ 时开关闭合，求 $t > 0$ 时的电感电流 $i(t)$ 和电压源

的电流 $i_S(t)$。

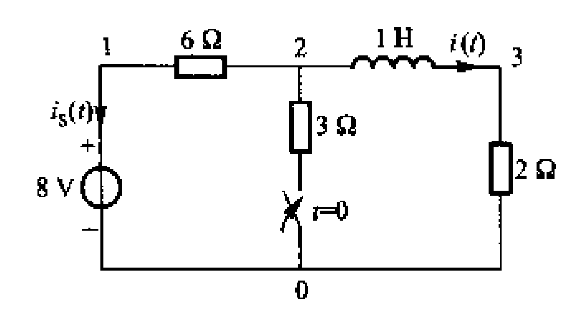


图 7-5 例 7-3 电路

解 $t=0$ 时的电感电流为

$$i(0)=\frac{8}{6+2}\text{A}=1\,\text{A}$$

设电感电流为 il,M 文件如下:

```
ckt = {'vs 1 0 8'
       'r 1 2 6'
       'r 2 0 3'
       'i 2 3 il'
       'r 3 0 2'};
sana(ckt);
eq1 = char('Dil' - (v2 - v3));
il = dsolve(eq1,'il(0) = 1')
ivs = subs(ivs,'il',il)
```

计算结果为

$$i(t)=\left(\frac{2}{3}+\frac{1}{3}\text{e}^{-4t}\right)\text{A}$$

$$i_S(t)=\left(-\frac{10}{9}-\frac{1}{9}\text{e}^{-4t}\right)\text{A}$$

利用 ezplot 函数可方便地绘制出符号表达式的波形。例如要在 $0<t<1\text{s}$ 范围内绘制电感电流的变化曲线,只要键入

```
ezplot(il,[0,1])
```

在图形窗口可得图 7-6 所示的曲线。

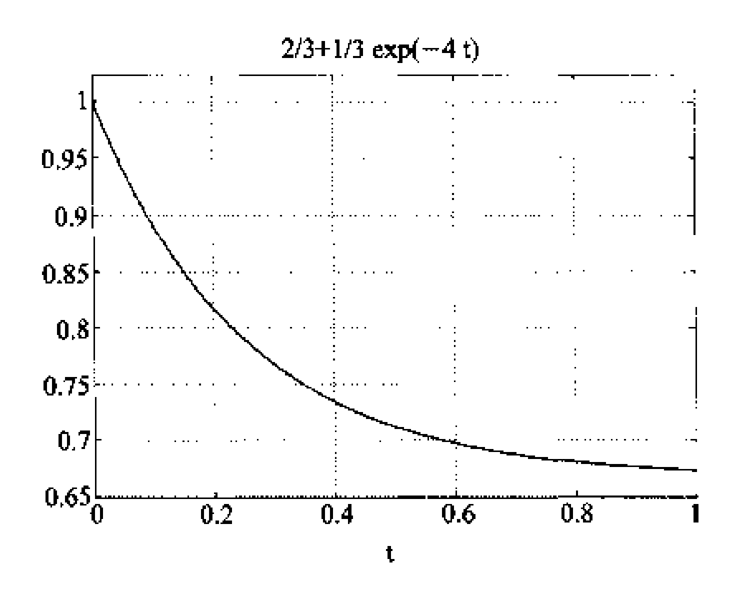


图 7-6 电感电流的波形

7.2 时域符号分析的一般方法

参看图 7-7(a)，设电路含有一个电容和一个电感，其余部分由电阻、受控源和独立电源组成。若用电压源 v_C 替代电容，用电流源 i_L 替代电感，电路如图 7-7(b)所示。假设图 7-7(b)所示电路具有唯一解，则用电阻电路的求解方法就可求出电容电流 i_C 和电感电压 v_L，它们均为 v_C 和 i_L 的函数

$$i_C = f_1(t, i_L, v_C)$$

$$v_L = f_2(t, i_L, v_C)$$

依据电容和电感元件的 VCR，则有

$$\begin{cases} C\dfrac{\mathrm{d}v_C}{\mathrm{d}t} = f_1(t, i_L, v_C) \\ L\dfrac{\mathrm{d}i_L}{\mathrm{d}t} = f_2(t, i_L, v_C) \end{cases} \tag{7-6}$$

式(7-6)构成以 v_C 和 i_L 为变量的联立微分方程组，利用初始值 $v_C(0)$ 和 $i_L(0)$ 就可求出 v_C 和 i_L，这两个量求出后，根据图 7-7(b)还可求出其他电压和电流。

编写程序的步骤如下：(1)用符号表示电容电压和电感电流，电容用电压源替代，电感用电流源替代，并用函数 sana 求解；(2)建立式(7-6)所示形式的微分方程，用函数 dsolve 求出电容电压和电感电流；(3)利用以上两个过程的解答再求解其他电压和电流。

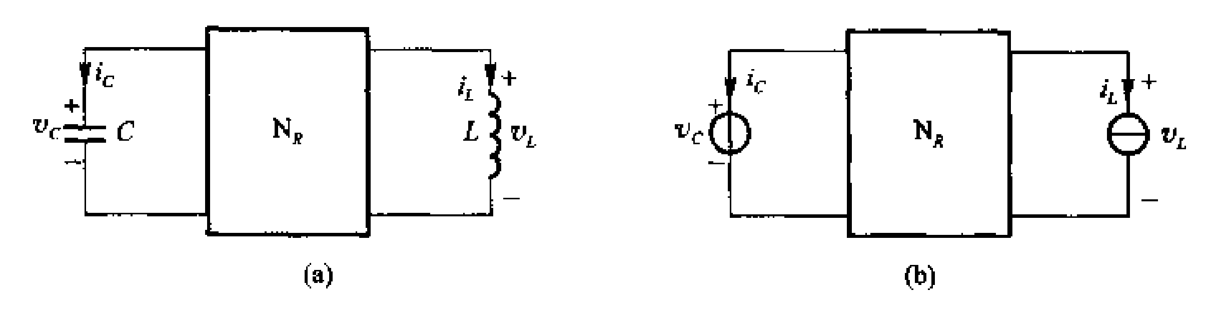


图 7-7 含有一个电感和一个电容的二阶电路

例 7-4 二阶电路的时域符号分析

电路如图 7-8 所示，已知 $v_C(0)=0\text{V}$，$i_L(0)=1\text{A}$，求结点电压和电容电流。

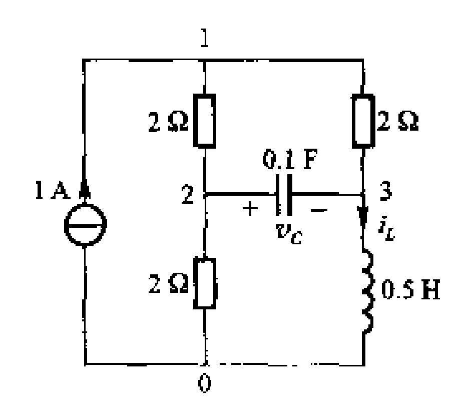


图 7-8 例 7-4 电路

解 M 文件如下：

```
ckt = {'is 0 1 1'
       'r 1 2 2'
       'r 2 0 2'
       'r 1 3 2'
       'vc 2 3 vc'
       'il 3 0 il'};
y = sana(ckt);                                          % ---(1)
eq1 = char('0.1 * Dvc' - ivc);                          % ---(2)
eq2 = char('0.5 * Dil' - v3);                           % ---(3)
[vc,il] = dsolve(eq1,eq2,'vc(0) = 0','il(0) = 1');      % ---(4)
y = subs(y,{'vc','il'},{vc,il});                        % ---(5)
y = simple(y);                                          % ---(6)
y = vpa(y,3)                                            % ---(7)
```

其中，语句(1)将电阻电路的解赋给 y，y 是由结点电压和电容电流组成的列向量；语句(2)和(3)建立以电容电压和电感电流为变量的微分方程；语句(4)用微分方程的求解指令求出电容电压 vc 和电感电流 il；语句(5)将电容电压和电感电流的解答代入输出向量 y 中；语句(6)对 y 中的各表达式进行简化；语句(7)以 3 位有效数字显示结果。

结果如下：

```
v1 = -2. * il + 3. - .5000 * vc
v2 = -2. * il + 2.
v3 = -1. * vc - 2. * il + 2.
ivc = -.2500 * vc + il - .5000
y =
[1.33 + .887 * exp( -3.25 * t) * sin(4.40 * t) - .333 * cos(4.40 * t) *
exp( -3.25 * t)]
[.667 + .641 * exp( -3.25 * t) * sin(4.40 * t) - .667 * cos(4.40 * t) *
exp( -3.25 * t)]
[      .107e - 2 * exp( -3.25 * t) * (827. * sin(4.40 * t) - 311. *
cos (4.40 * t))]
[      .107e - 2 * exp( -3.25 * t) * (827. * sin(4.40 * t) - 311. *
cos (4.40 * t))]
```

例 7-5 时域符号分析

电路如图 7-9 所示，已知 $V=6$ V，$R_1=2\Omega$，$R_2=6\Omega$，$L_1=1$H，$L_2=4$H，开关在 $t=0$ 时闭合，求电感电流和电阻 R_1 的电压。

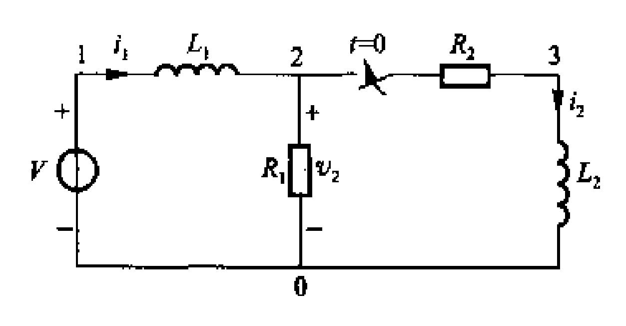


图 7-9 例 7-5 电路

解 在 $t=0$ 时刻的电感电流为

$$i_1(0)=\frac{V}{R_1}=3\text{A},\qquad i_2(0)=0\text{A}$$

M 文件如下：

```
ckt = {'v 1 0 6'
       'i1 1 2 i1'
       'r1 2 0 2'
       'r2 2 3 6'
       'i2 3 0 i2'};
sana(ckt);
eq1 = char('Di1' - (v1 - v2));
eq2 = char('4 * Di2' - v3);
[i1,i2] = dsolve(eq1,eq2,'i1(0) = 3','i2(0) = 0')
v2 = subs(v2,{'i1','i2'},{i1,i2})
```

执行程序后有

```
i1 =   1/2 * exp( -3 * t) -3/2 * exp( -t) +4
i2 =   -3/4 * exp( -t) -1/4 * exp( -3 * t) +1
v2 =   3/2 * exp( -3 * t) -3/2 * exp( -t) +6
```

在采用以上方法对电路进行时域符号分析时，要求电容和电感用电源替代以后电路必须具有唯一解。当电路不满足这一条件时，需要作特殊处理，此处从略。

高阶电路的求解方法与二阶电路的类似。

7.3 初值常微分方程问题的数值求解

动态电路（或系统）的方程通常能够写成如下形式：

$$\begin{bmatrix} y_1' \\ y_2' \\ \vdots \\ y_N' \end{bmatrix} = \begin{bmatrix} f_1(t,y_1,y_2,\cdots,y_N) \\ f_2(t,y_1,y_2,\cdots,y_N) \\ \vdots \\ f_N(t,y_1,y_2,\cdots,y_N) \end{bmatrix} \tag{7-7}$$

其中，t 是时间；y_k 为第 k 个方程变量，$k=1,2,\cdots,N$；y_k' 是 y_k 的一阶导数；$f_k(\,)$ 为数学函数。为简明起见，令 $\boldsymbol{y}$ 为所有 y_k 组成的列向量，$\boldsymbol{f}(t,\boldsymbol{y})$ 为函数 $f_k(\,)$ 组成的列向量，则式(7－7)又可表示为

$$\boldsymbol{y}' = \boldsymbol{f}(t,\boldsymbol{y}) \tag{7-8}$$

式(7－8)称为系统的常微分方程，或状态方程，$\boldsymbol{y}$ 为状态向量。当初值 $\boldsymbol{y}(t_0)$ 给定时，根据式(7－8)和 $\boldsymbol{y}(t_0)$ 求解 $\boldsymbol{y}(t)$ 称为初值常微分方程问题。如果用数值

方法求解式(7-8),只能求解一些离散时间处 y 的近似值,即所谓数值解。

对动态电路,在建立式(7-8)所示的微分方程时,一般选电容电压和电感电流为变量。然而,对复杂电路,要用系统性的方法写出式(7-8)所示形式的方程却比较困难,这是因为最容易建立的结点方程并不具备式(7-8)所示形式,只有对结点方程进行必要的运算并消去其他变量后,才有可能得到式(7-8)所示方程。尽管如此,手工建立简单电路的方程还是比较容易,这样,就可以直接利用数学中已有的许多求解方法进行电路分析。

在计算数学中,式(7-8)的求解有多种算法①,每种算法都有其特殊性,至今还没有一个普遍适用的能够高效计算所有问题的方法。MATLAB 软件充分考虑了这种情况,它提供了 7 种计算函数,用户可以根据所分析的具体问题选用。

在使用某种算法求解的过程中,某一步产生的误差在以后时间的求解中如果不能逐步削弱,一般来说这种方法或选定的计算步长就不宜采用。在研究这一问题时,通常假设微分方程为

$$y' = \lambda y$$

其中,λ 与时间 t 和方程变量 y 无关。设 $\bar{h}=\lambda h$,h 为计算步长,$h=t_{n+1}-t_n$。对某一算法,只有当 $\bar{h}$ 满足一定条件时误差才会逐步减小,相应 $\bar{h}$ 的集合称为算法的稳定区域。稳定区域越广,说明这种方法的稳定性就越好。对同一种算法,算法的阶次越高,稳定区域就越小。

对高阶微分方程,特征值实部绝对值的大小直接影响计算步长的选取,取值过小,总的计算时间变长,取值过大,对特征值实部绝对值比较大的响应分量可能在计算结果中无法反映。若定义特征值实部绝对值的最大值与最小值的比为刚性,刚性越大,对计算步长的要求越苛刻,刚性很大的方程称为刚性方程。

在 MATLAB 中,非刚性方程的求解函数有三种。

ode45 函数采用 4 阶、5 阶 Runge-Kutta 算法,它是一种变步长的单步方法。一般来说,ode45 是求解普通微分方程的首选函数。

ode23 函数基于 2 阶、3 阶 Runge-Kutta 公式,它也是一种单步方法。由于该算法的阶次低于 ode45 的阶次,在同样的计算精度条件下,ode23 只能取较小的计算步长。也正由于这个原因,它处理中度刚性方程的能力比 ode45 强。

ode113 函数采用预报校正方法计算。该算法只能用于光滑系统的分析,由于其导数的计算量小,当精度要求较高时,ode113 要比 ode45 更有效。如果使用 ode45 的计算时间比较长,可改用 ode113 试算。ode113 是一种多步方法。

---

① 见 7.5 节内容或计算方法书。

刚性方程的求解函数有四种。

ode15s 函数采用变阶 Gear 算法，最高阶次可取到 5。该算法适合于刚性方程的求解，对非刚性方程的计算效率偏低。

ode23s 函数基于改进的 2 阶 Rosenbrock 公式，属单步方法，在低精度要求时它有可能比 ode15s 有效。对某些刚性问题，它比 ode15s 具有更快的计算速度。

ode23t 函数采用梯形算法，适合中度刚性问题的计算。

ode23tb 函数采用隐式 Runge - Kutta 公式，如同 ode23s，在低精度要求时它有可能比 ode15s 有效。

在 MATLAB 中，所有求解常微分方程的函数使用相同的语句格式，这使得采用不同算法分析同一个问题非常便利，只需要简单的改变函数名称即可。对微分方程的所有求解函数，最简单的语句格式为

```
[t,y] = solver(odefun,tspan,y0)
```

其中，

(1) solver 表示上述任一个求解函数，如 ode45，ode15s，实际计算时，上面语句中的 solver 要使用具体的微分方程求解函数。

(2) 变元 odefun 表示计算 $f(t,y)$ 的函数，该函数的格式如下：

```
fty = odefun(t,y,p1,p2,...)
```

其中，p1，p2 等用于给 odefun 传递参数；y 为方程变量的列向量；函数体按式(7-7)等号右方的表达式编写；函数返回 $f(t,y)$ 的值。

(3) 变元 tspan 是指定的积分区间向量，tspan(1) 为起始时间，初始值是方程变量在该时间的值。如果 tspan 只有两个元素[t0 tf]，程序将返回每一积分步长处的值，当 tspan 多于两个元素时，程序给出指定时间处的值。时间值必须按时间以递增或递减的顺序排列，tspan 给定的离散时间不会影响程序内部各时间点的计算步长。

(4) y0 是初始值列向量。

(5) 输出变元 t 是离散时间的列向量；y 是存放计算结果的二维数组，其行对应于时间，列对应于 y 中的变量。

求解微分方程更为完整的语句格式为

```
[t,y] = solver(odefun,tspan,y0,options,p1,p2...)
```

其中，第 4 个输入变元 options 是设置积分特性的结构体对象，有关参数的设置将在下一节介绍。如果计算时全部采用程序设置的隐含值，而又要给 solver 传递参数 p1、p2 等，则要令 options = [ ] 。

例 7-6 van der Pol 方程

在无线电技术研究中,van der Pol 于 1926 年提出重要的非线性现象——自激振荡。能量保守的自由振荡只是一种理想化模型,在实际系统中总会由于阻尼而振荡衰减下来。自激振荡阐明了一个系统怎样依靠内部的常能量源而维持振荡的,van der Pol 方程如下式所示:

$$y'' - \mu(1-y^2)y' + y = 0 \tag{7-9}$$

若$\mu=1, y(0)=2, y'(0)=0$,计算终止时间 $t_f=20$,求 $y(t)$。

解 若令 $y_1=y, y_1'=y_2$,式(7-9)可写成

$$y_2' = \mu(1-y^2)y_2 - y_1$$

用状态方程表示式(7-9),有

$$\begin{cases} y_1' = y_2 \\ y_2' = \mu(1-y^2)y_2 - y_1 \end{cases}$$

M 文件如下:

```
function b206
[t,y] = ode45(@vdp1,[0 20],[2;0]);
plot(t,y(:,1),'-',t,y(:,2),'--')
title('Solution of van der Pol Equation, \mu = 1');
xlabel('t');
ylabel('solution y');
legend('y_1','y_2')

function fty = vdp1(t,y)
mu = 1;
fty = [y(2); mu * (1 - y(1)^2) * y(2) - y(1)];
```

其中,mu 表示μ。y_1 和 y_2 的曲线如图 7-10 所示。

如果取$\mu=1\ 000, t_f=3\ 000$,用 ode45 函数计算会发现所用的计算时间很长,这是由方程的刚性造成的,而当改用 ode15s 进行计算,则效率明显提高。

例 7-7 电路的状态空间分析

电路如图 7-11 所示,已知 $R_1=100\Omega, R_2=2\Omega, R_3=30\Omega, C=0.1\ \mu F, L=0.1 H, v_S=\cos(1\ 000t)\varepsilon(t)$ V,求结点电压 v_1。

解 首先列出电路的微分方程。用电压源替代电容,用电流源替代电感,如图 7-12 所示。利用结点法或叠加定理求结点 1 的电压,有

$$v_1 = \frac{R_2 v_S + R_1 v_C - R_1 R_2 i_L}{R_1 + R_2}$$

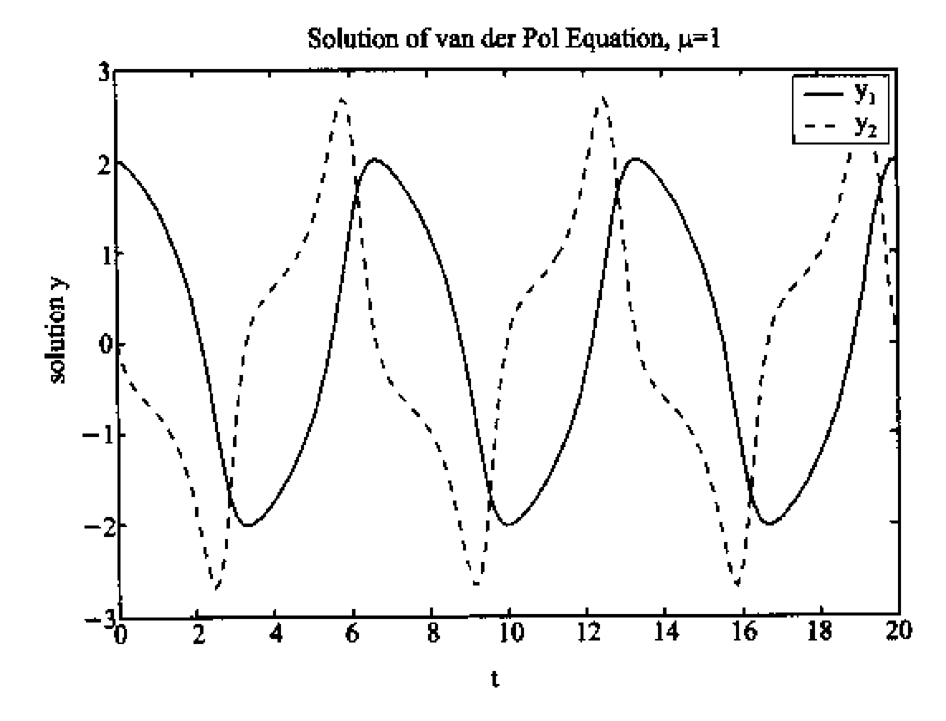


图 7－10 van der Pol 方程的解

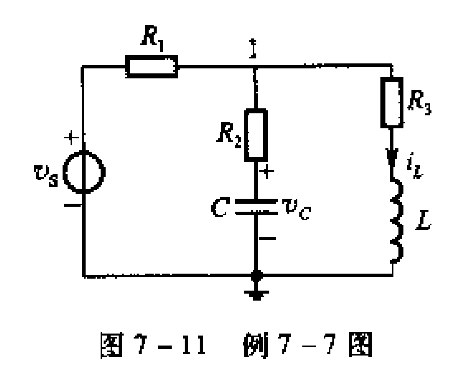


图 7－11 例 7－7 图

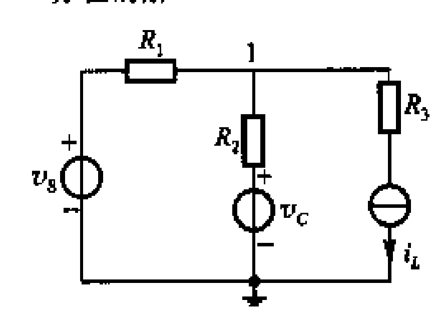


图 7－12 动态元件的独立电源替换

流经电容的电流和电感上的电压分别为

$$C\frac{\mathrm{d}v_C}{\mathrm{d}t}=\frac{1}{R_2}(v_1-v_C)$$

$$L\frac{\mathrm{d}i_L}{\mathrm{d}t}=v_1-R_3i_L$$

根据已知条件，$t<0$ 时电源电压为零，故 $v_C(0)=0$，$i_L(0)=0$，设计算的终止时间 $t_f=0.03\mathrm{s}$，使用 ode45 函数求解，M 文件如下：

```
function b207
t = [0:1e-5:0.02];
[t,y] = ode45(@ckt,t,[0;0]);
vs = cos(1000 * t);
r1 = 100;r2 = 2;
```

```
v1 = (r2 * vs + r1 * y(:,1) - r1 * r2 * y(:,2))/(r1 + r2);
plot(t,v1)
xlabel('t');

function fty = ckt(t,y)
r1 = 100;r2 = 2;r3 = 30;c = 0.1e - 6;l = 0.1;
vs = cos(1000 * t);
v1 = (r2 * vs + r1 * y(1) - r1 * r2 * y(2))/(r1 + r2);
fty = [(v1 - y(1))/r2/c
       (v1 - r3 * y(2))/l];
```

电压 v_1 的波形如图 7－13 所示。

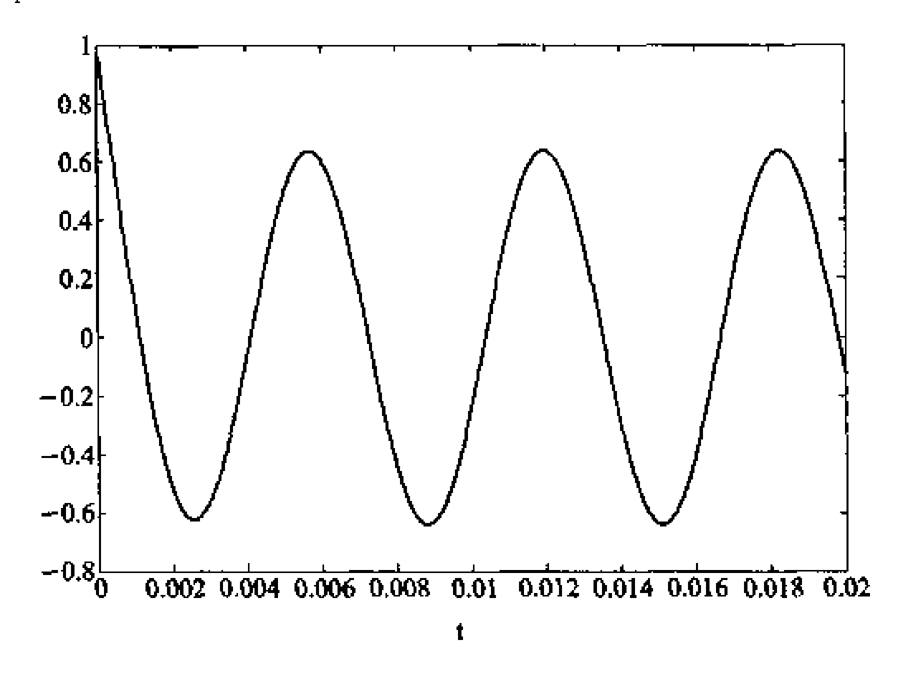


图 7－13 电压 v_1 的波形

例 7－8 蔡氏混沌电路

蔡氏(Chua L. O.)混沌电路如图 7－14 所示，它由四个线性器件(r_L 为电感线圈的寄生电阻)和一个非线性电阻组成。在特定的参数值及初始条件下，以状态变量为坐标的曲线具有双涡卷形状，因而也称此电路为双涡卷电路。已知 $R = 1.001\text{k}\Omega, C_2 = 178.6\text{nF}, L = 12.46\text{mH}, r_L = 20.2\Omega, C_1 = 16.667\text{nF}$，非线性电阻的特性由三段直线构成，如图 7－15 所示。

$$i_A = g_b v_A + 0.5(g_a - g_b)\{|v_A + b_p| - |v_A - b_p|\}$$

式中，$g_a = -1.129\text{mS}, g_b = -0.702\ \text{mS}, b_p = 1\ \text{V}$。试以 v_2、v_1、i_L 为变量，绘制三

维曲线。

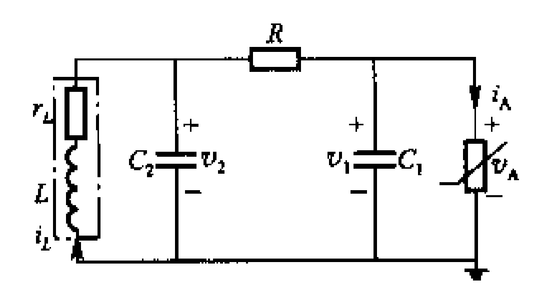


图 7－14 蔡氏混沌电路

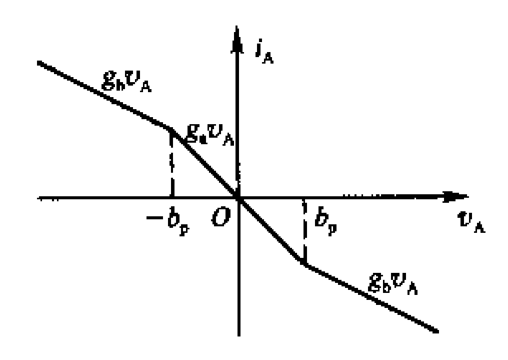


图 7－15 非线性电阻的特性

解 蔡氏电路的微分方程如下：

$$C_1 \frac{dv_1}{dt} = \frac{1}{R}(v_2 - v_1) - i_A$$

$$C_2 \frac{dv_2}{dt} = \frac{1}{R}(v_1 - v_2) + i_3$$

$$L \frac{di_L}{dt} = -v_2 - R_L i_L$$

其中，

$$i_A = g_b v_1 + 0.5(g_a - g_b)\{|v_1 + b_p| - |v_1 - b_p|\}$$

由于图 7－14 所示电路不包含随时间变化的输入（称为自治系统），在时域分析时必须给定一个非零值的初始值，设取 $v_1(0) = 0.5$ V。M 文件如下：

```
function b208
t = [0, 0.1];
[t,y] = ode45(@chua,t,[0.5;0;0]);
plot3(y(:,2),y(:,1),y(:,3))
xlabel('v2'),ylabel('v1'),zlabel('iL')
```

```
grid

function fty = chua(t,y)
r = 1001;c2 = 178.6e-9;l = 12.46e-3;
ga = -1.129e-3;gb = -0.702e-3;bp = 1;
rl = 20.2;
c1 = 16.667e-9;
ia = gb*y(1) + 0.5*(ga - gb)*(abs(y(1) + bp) - abs(y(1) - bp));
fty = [((y(2) - y(1))/r - ia)/c1
      ((y(1) - y(2))/r + y(3))/c2
      (-y(2) - rl*y(3))/l];
```

程序绘制的三维曲线如图 7-16 所示。

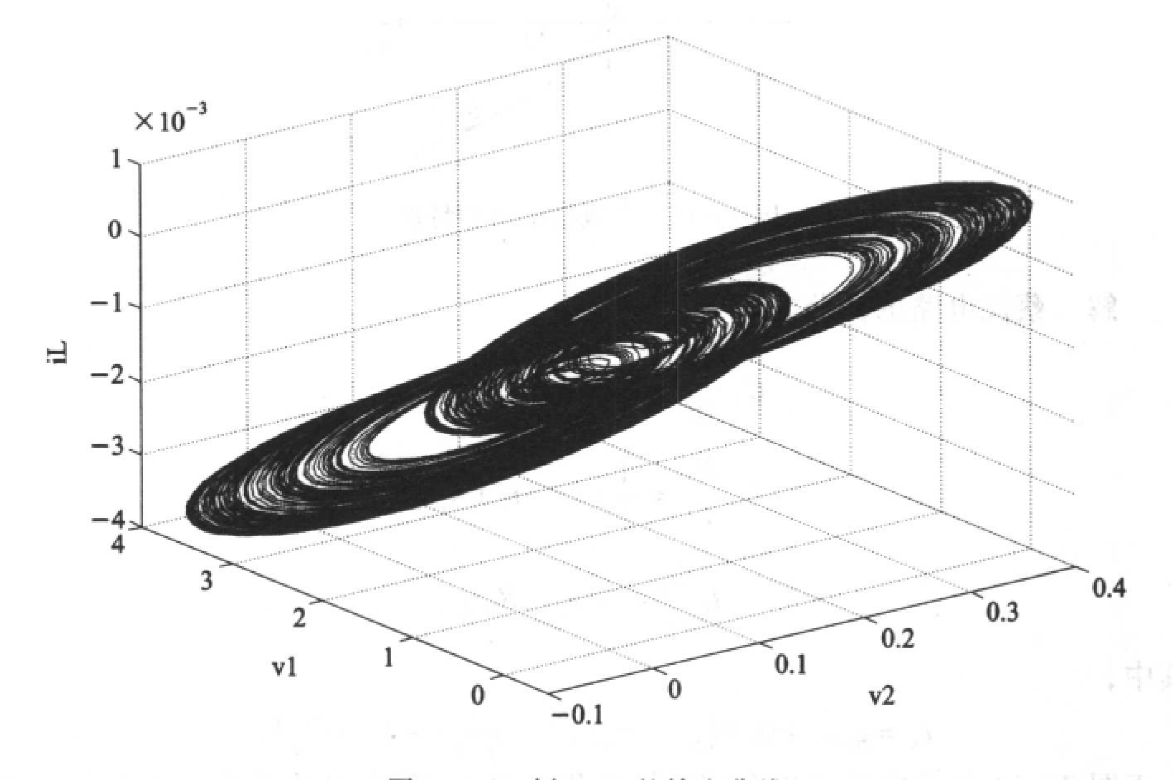


图 7-16　例 7-8 的输出曲线

7.4　微分方程计算函数的属性设置

动态系统更一般的方程形式为：

$$\boldsymbol{M}(t,\boldsymbol{y})\boldsymbol{y}' = \boldsymbol{f}(t,\boldsymbol{y}) \tag{7-10}$$

其中,$\boldsymbol{M}(t,\boldsymbol{y})$是$N$阶方阵,称为质量矩阵,它可以是常数矩阵,也可以是$t$和$\boldsymbol{y}$的函数。当$\boldsymbol{M}(t,\boldsymbol{y})$的逆矩阵存在时,式(7-10)可化为式(7-8)形式。当$\boldsymbol{M}(t,\boldsymbol{y})$奇异时,某些变量之间仅仅具有代数约束关系,式(7-10)为微分代数方程。

当质量矩阵非奇异时,7.3节介绍的7个求解函数也能用于式(7-10)。但当质量矩阵奇异时,只能使用ode15s和ode23t求解。

7.3节已指出,在调用微分方程的计算函数时,通过给定可选项变元可以改变程序的隐含积分属性,调用格式如下:

```
[t,y] = solver(odefun,tspan,y0,options)
```

其中,可选项变元options为结构体对象,用odeset指令设置,其格式为

```
options = odeset('name1',value1,'name2',value2,...)
```

其中,name1、name2等表示要设置的参数名称,value1、value2等为对应参数的值,这些值将替代程序内部设定的隐含值。与质量矩阵及微分代数方程有关的参数如表7-1所示。

当使用ode15s求解微分代数方程时,有时需要限制算法的阶次,表7-2给出了有关参数的设置。

在每步计算中,计算步长根据精度要求由程序自动确定。但有时却需要人工干预,表7-3给出了有关参数。

表7-1 与质量矩阵及微分代数方程有关的参数

参数名称	属性值	说　明
Mass	质量矩阵的矩阵名或函数名	当质量矩阵为常数矩阵时,给出矩阵名。当质量矩阵与时间和/或方程的变量有关时,给出有关函数名(@Mfun)。这时,质量矩阵用函数(Mfun)描述。ode23s只能求解常数质量矩阵
MStateDependence	None weak strong	反映质量矩阵是否与方程变量y有关的参数。None意味着质量矩阵只与t有关,而weak和strong指质量矩阵与方程变量y有关,weak为程序的隐含值,这时程序使用逼近方法求解代数方程,但ode23s除外
MvPattern	稀疏矩阵名	当雅可比矩阵是稀疏阵时,设置其稀疏性,详见Help
MassSingular	Yes no maybe	质量矩阵的奇异性设置,只能用于ode15s和ode23t函数。当质量矩阵非奇异时,属性值为no。属性的隐含值为maybe,这时,质量矩阵的奇异性由程序检测。如果微分代数方程$y(t_0)$和$y'(t_0)$在起始时间不满足一致性,程序则把它们作为猜测值来计算最为接近的一致性初始值
InitialSlope	$\boldsymbol{y}'(t_0)$列向量零值向量	一致性初始斜率值$\boldsymbol{y}'(t_0)$,即$M(t_0,\boldsymbol{y}(t_0))\boldsymbol{y}'(t_0)=f(t_0,\boldsymbol{y}(t_0))$,只能用于ode15s和ode23t函数

表 7-2 ode15s 的属性设置

参数名称	属性值	说明
MaxOrder	1 2 3 4 5	ode15s 为变阶求解程序，该参数指定计算中所使用的最大阶次，隐含值为 5
BDF	on off	值为 on 时，ode15s 改用称为 Gear 方法的后向差分公式(BDFs)计算，以取代隐含的数值差分公式（NDFs）。通常，NDFs 更为有效。1 阶和 2 阶 BDFs 和 NDFs 都是 A-稳定的

表 7-3 计算步长与误差控制的属性设置

参数名称	属性值	说明
MaxStep		允许使用的最大计算步长，隐含值为计算区间的十分之一
InitialStep		建议使用的初始计算步长，如果计算结果的误差太大，程序将改用较小的步长
AbsTol	1e-6	绝对误差。如果为标量，该误差应用于所有方程变量。AbsTol 也可以为指定各个变量绝对误差的向量
RelTol	1e-3	相对误差，当值大于 AbsTol 给定的绝对误差时，它直接影响计算结果的有效位数

例 7-9 全耦合电感

图 7-17 所示电路含有耦合电感，已知 $V=5\text{V}$，$R_1=R_2=1\Omega$，$L_1=4\text{H}$，$L_2=1\text{H}$，$M=2\text{H}$，$i_1(0_-)=i_2(0_-)=0\text{A}$，求 $t>0$ 时的电流 $i_1(t)$ 和电压 $v_2(t)$。

解 以 i_1 和 i_2 为变量对电路中的两个回路分别应用 KVL，有

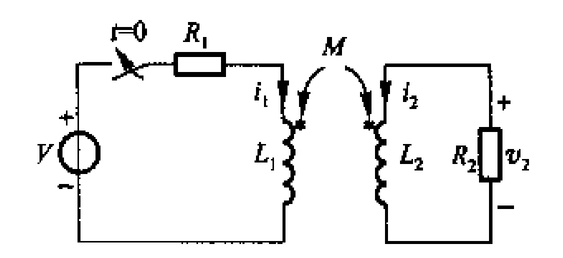


图 7-17 含有耦合电感的电路

$$\begin{cases} L_1 \dfrac{\mathrm{d}i_1}{\mathrm{d}t} + M \dfrac{\mathrm{d}i_2}{\mathrm{d}t} = V - R_1 i_1 \\ M \dfrac{\mathrm{d}i_1}{\mathrm{d}t} + L_2 \dfrac{\mathrm{d}i_2}{\mathrm{d}t} = -R_2 i_2 \end{cases}$$

由于 $k = \dfrac{M}{\sqrt{L_1 L_2}} = 1$,即电感全耦合,因此,上式中的质量矩阵奇异,故上式为电路的微分代数方程。输出电压

$$v_2 = -R_2 i_2$$

M 文件如下:

```
function b209
% mass matrix
l1 =4; l2 =1; m =2;
M = [l1, m;m, l2];
% initial condition
y0 = [0;0];
tspan = linspace(0,20,201);
options = odeset('Mass',M);
[t,y] = ode15s(@f,tspan,y0,options);
v2 = -y(:,2);
plot(t,y(:,1),t,v2)
% --------------------
function out = f(t,y)
vs =5; r1 =1; r2 =1;
out = [vs - r1 * y(1)
       -r2 * y(2)];
```

电流 $i_1(t)$ 和电压 $v_2(t)$ 的波形如图 7－18 所示①，由图可见，$i_1(0_+)=1\ \mathrm{A}\neq i_1(0_-)$，即电感电流在 $t=0$ 处不连续。

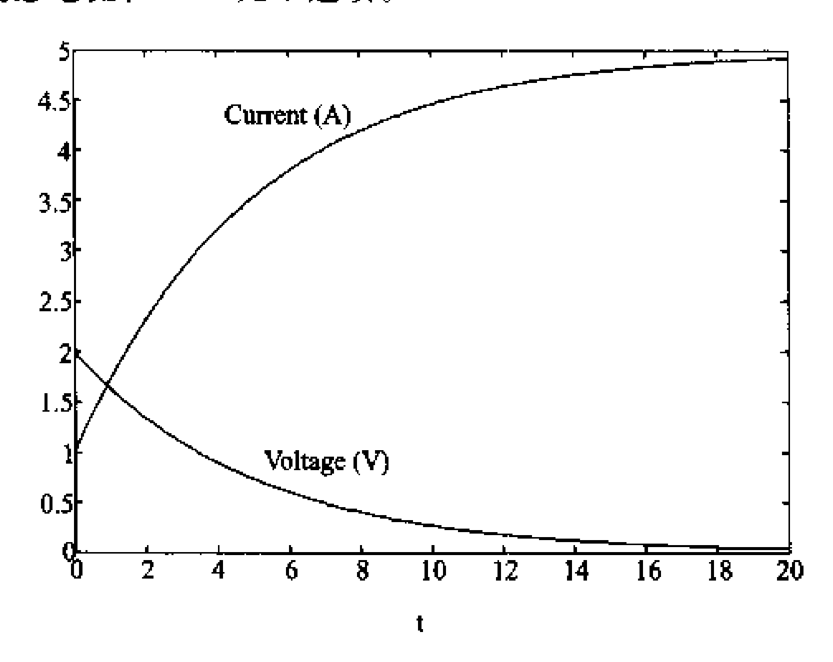


图 7－18 例 7－9 的输出波形

7.5 离散时间电路模型

电路时域数值分析有的采用状态方程法，有的采用结点方程法。前者的突出优点是：计算数学中有比较多的求解方法可以利用，如常用的 Runge－Kutta 方法。然而，电路状态方程的建立过程较为复杂，至今还没有一个十分令人满意的方法。在电子电路分析中，一般采用结点法。

本节介绍常微分方程的一些计算方法及其在电路计算中的应用。

单变量微分方程可表示为

$$\frac{\mathrm{d}x}{\mathrm{d}t}=f(t,x) \tag{7-11}$$

设离散时间间隔为 h，$x(t)$ 在离散时间 $t=t_n$ 处的计算值为 x_n（n 取整数），$f_n=f(t_n,x_n)$。设在 t_{n+1} 之前的计算值等于真解，把 $x(t_{n+1})$ 和 $x'(t_{n+1})$ 在 $t=t_n$ 处用泰勒级数展开，有

① 由于在图形窗口中可利用下拉式菜单对图形进行编辑和添加注解，为节省纸张，在本例及后续一些程序中将省去有关指令。

$$x(t_{n+1}) = x_n + h\,x'_n + \frac{1}{2!}h^2 x''_n + \cdots \qquad (7-12a)$$

$$x'(t_{n+1}) = x'_n + hx''_n + \cdots \qquad (7-12b)$$

从以上两式中消去 x'_n，得

$$x(t_{n+1}) = x_n + hx'(t_{n+1}) - \frac{1}{2}h^2 x''_n + \cdots \qquad (7-13)$$

上式中如果只取到 h 的线性项，则 $x(t_{n+1})$ 可按下式近似计算

$$x_{n+1} = x_n + hf_{n+1} \qquad (7-14)$$

式(7－14)称为求解微分方程的后向欧拉公式，根据式(7－13)，其局部截断误差近似为

$$E \approx -\frac{1}{2}h^2 x''_n$$

它与 h^2 成正比，在忽略舍入误差的情况下，h 愈小，则计算结果愈接近真解。当 $f(t,x)$ 与 x 成线性关系时，从式(7－14)能够导出求解 x_{n+1} 的递推公式，根据初始条件可逐步求得 $x(t)$ 在各离散时间处的近似值。当 $f(t,x)$ 与 x 成非线性关系时，根据 x_n 求 x_{n+1} 需要使用迭代方法。

式(7－14)在求解 x_{n+1} 时只用到 x_n，是一种单步方法。为了充分利用前几步计算得到的信息，也可用 t_{n+1} 之前离散值的线性组合求解，在相同时间间隔情况下可以提高计算的准确度，这种方法称为线性多步法。例如，取前两个时间点时，x_{n+1} 可表示为

$$x_{n+1} = \alpha_1 x_n + \alpha_2 x_{n-1} + h(\beta_0 f_{n+1} + \beta_1 f_n + \beta_2 f_{n-1})$$

设取 $\beta_1 = \beta_2 = 0$，有

$$x_{n+1} = \alpha_1 x_n + \alpha_2 x_{n-1} + h\beta_0 f_{n+1} \qquad (7-15)$$

其局部截断误差

$$E = x(t_{n+1}) - \alpha_1 x_n - \alpha_2 x_{n-1} - h\beta_0\, x'_{n+1}$$

在 t_n 处用泰勒级数展开，可得

$$E = (1-\alpha_1-\alpha_2) + h[1+\alpha_2-\beta_0]x'_n + h^2\left[\frac{1}{2!}(1-\alpha_2) - \beta_0\right]x''_n +$$

$$h^3\left[\frac{1}{3!}(1+\alpha_2) - \frac{1}{2!}\beta_0\right]x'''_n + \cdots$$

适当选取 α_1、α_2、β_0 使 h^0、h^1、h^2 项的系数为零，即

$$\begin{cases} 1-\alpha_1-\alpha_2 = 0 \\ 1+\alpha_2-\beta_0 = 0 \\ \dfrac{1}{2!}(1-\alpha_2) - \beta_0 = 0 \end{cases}$$

求得

$$\alpha_1=\frac{4}{3},\quad \alpha_2=-\frac{1}{3},\quad \beta_0=\frac{2}{3}$$

将各系数代入式(7－15),有

$$x_{n+1}=\frac{1}{3}(4x_n-x_{n-1})+\frac{2}{3}hf_n \tag{7-16}$$

上式称为求解微分方程的二阶 Gear 公式(后向欧拉公式也是一阶 Gear 公式),其局部截断误差

$$E\approx-\frac{2}{9}h^3x'''_n$$

它与 h^3 成正比。

由于式(7－16)在求 x_{n+1} 时要用到前两个离散时间处的值,当微分方程的初值 $x(0)$ 给定后,它不能自行起步,解决的方法是,以 $h/2$ 为时间间隔,用式(7－14)求出 $x(h/2)$,再用式(7－16)求出 $x(h)$,以后各离散时间处按式(7－16)求解。

同理可求出 3 阶、4 阶 Gear 公式

$$x_{n+1}=\frac{1}{11}(18x_n-9x_{n-1}+2x_{n-2}+6hf_{n+1})$$

$$x_{n+1}=\frac{1}{25}(48x_n-36x_{n-1}+16x_{n-2}-3x_{n-3}+12hf_{n+1})$$

在实际使用中,考虑到方法的稳定性问题和舍入误差的影响, Gear 公式的阶次最高取到 6 阶,高于 6 阶的积分方法不一定能取得比较好的精度。

例 7－10　后向欧拉法与二阶 Gear 法的比较

给定微分方程

$$\begin{cases}\dfrac{\mathrm{d}x(t)}{\mathrm{d}t}=-x+1\\ x(0)=0\end{cases} \tag{7-17}$$

设 $h=0.05$,在 $0<t\leqslant1$ 内用后向欧拉公式、二阶 Gear 公式求 $x(t)$。

解　(1) 后向欧拉法。

据式(7－14)和式(7－17)有

$$x_{n+1}=x_n+h(-x_{n+1}+1)$$

于是

$$x_{n+1}=\frac{x_n+h}{1+h} \tag{7-18}$$

(2) 二阶 Gear 法。

由式(7-16)和式(7-17)得

$$x_{n+1}=\frac{1}{3}(4x_n-x_{n-1})+\frac{2h}{3}(-x_{n+1}+1)$$

则

$$x_{n+1}=\frac{4x_n-x_{n-1}+2h}{3+2h} \tag{7-19}$$

式(7-17)的解析解为

$$x(t)=1-e^{-t} \tag{7-20}$$

表7-4给出了式(7-18)~式(7-20)的计算结果,显然,二阶 Gear 法具有比较好的计算精度。

表7-4 计算结果的比较

时间	后向欧拉法	二阶 Gear 法	解析解
0.05	0.04762	0.04838	0.04877
0.10	0.09297	0.09469	0.09516
0.15	0.13616	0.13883	0.13929
0.20	0.17730	0.18084	0.18127
0.25	0.21647	0.22082	0.22120
0.30	0.25378	0.25885	0.25918
0.35	0.28932	0.29503	0.29531
0.40	0.32316	0.32944	0.32968
0.45	0.35539	0.36217	0.36237
0.50	0.38609	0.39331	0.39347
0.55	0.41532	0.42292	0.42305
0.60	0.44316	0.45109	0.45119
0.65	0.46968	0.47788	0.47795
0.70	0.49493	0.50337	0.50341
0.75	0.51898	0.52761	0.52763
0.80	0.54189	0.55067	0.55067
0.85	0.56370	0.5726	0.57259
0.90	0.58448	0.59346	0.59343
0.95	0.60427	0.61331	0.61326
1.00	0.62311	0.63218	0.63212

在电路的数值计算中,电容和电感的 VCR 具有式(7-11)所示形式,电容

元件的 VCR 可表示为

$$\frac{\mathrm{d}v}{\mathrm{d}t}=\frac{1}{C}i$$

对上式应用后向欧拉公式,有

$$v_{n+1}=v_n+\frac{h}{C}i_{n+1} \tag{7-21}$$

根据式(7-21)可给出电容的离散时间模型,见图 7-19。离散时间电路方程可利用离散时间电路模型来建立。

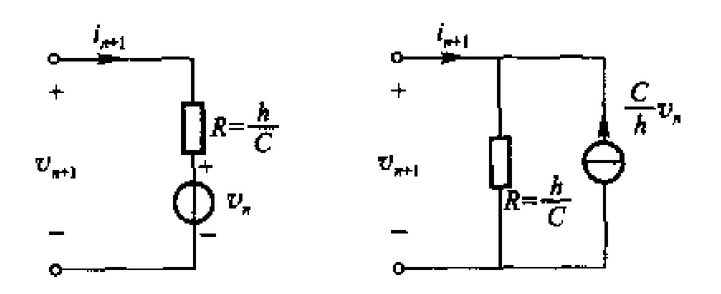


图 7-19　电容的离散时间模型(后向欧拉法)

若使用二阶 Gear 公式,电容元件的离散时间关系式为

$$v_{n+1}=\frac{1}{3}(4v_n-v_{n-1})+\frac{2h}{3C}i_{n+1} \tag{7-22}$$

与上式对应的离散时间模型如图 7-20 所示。

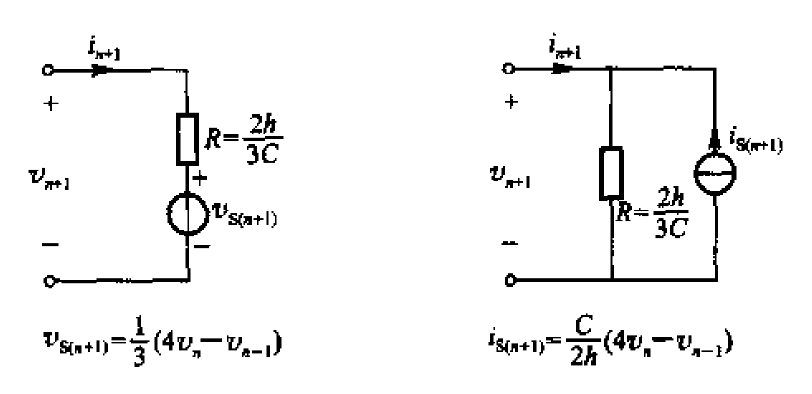


图 7-20　电容的离散时间模型(二阶 Gear 法)

电感元件的 VCR 为

$$\frac{\mathrm{d}i}{\mathrm{d}t}=\frac{1}{L}v$$

利用后向欧拉公式和二阶 Gear 公式分别得

$$i_{n+1}=i_n+\frac{h}{L}v_{n+1} \tag{7-23}$$

$$i_{n+1}=\frac{4i_n-i_{n-1}}{3}+\frac{2h}{3L}v_{n+1} \tag{7-24}$$

其离散时间模型如图 7－21 和图 7－22 所示。

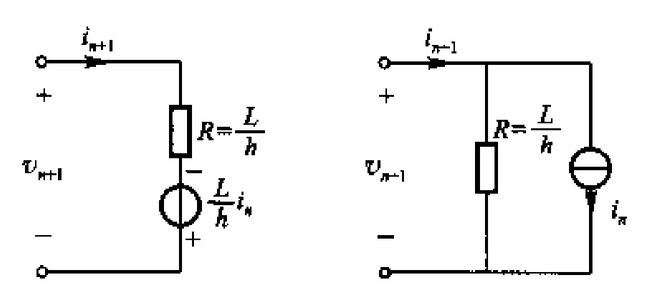


图 7－21 电感的离散时间模型（后向欧拉法）

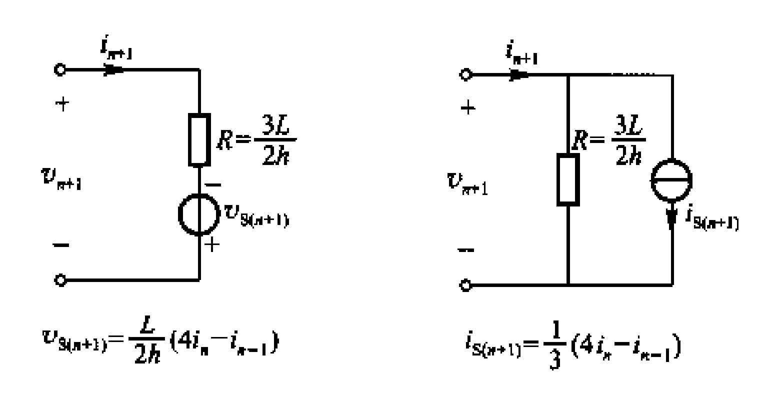


图 7－22 电感的离散时间模型（二阶 Gear 法）

电路的时域分析过程可表述为：(1)选取合适的积分算法，画出电路的离散时间模型；(2)利用给定的初始条件，建立离散时间模型的结点方程；(3)从方程求出电路的有关变量；(4)利用已求出的电容电压和电感电流重新建立结点方程并求解。由此可见，在每一离散时间处，电路的时域分析相当于对电路进行直流分析。

例 7－11 离散时间结点方程

电路如图 7－23 所示，已知 $R=1\ \Omega$，$C=0.5\ \text{F}$，$L=0.1\ \text{H}$，$i=\sin t$，初始值 $v_1(0)=1\text{V}$，$i_3(0)=0\text{A}$，试给出用后向欧拉法求解电路的离散时间结点方程。

解 离散时间电路模型如图 7－24 所示。如果把电感电流也作为方程变量，结点方程为

结点 1：$$\left(\frac{1}{R}+\frac{C}{h}\right)v_{1(n+1)}-\frac{1}{R}v_{2(n+1)}=i_{(n+1)}+\frac{C}{h}v_{1(n)}$$

结点 2：$$-\frac{1}{R}v_{1(n+1)}+\frac{1}{R}v_{2(n+1)}+i_{3(n+1)}=0$$

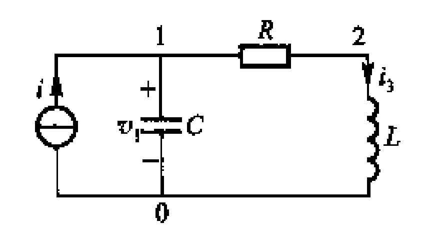


图 7－23 例 7－11 电路

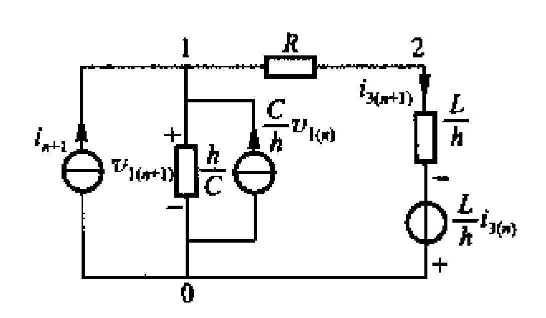


图 7－24 电路的离散时间模型

电感支路： $v_{2(n+1)}-\frac{L}{h}i_{3(n+1)}=-\frac{L}{h}i_{3(n)}$

代入已知数据得

$$\begin{bmatrix} 1+\frac{0.5}{h} & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & -\frac{0.1}{h} \end{bmatrix}\begin{bmatrix} v_{1(n+1)} \\ v_{2(n+1)} \\ i_{3(n+1)} \end{bmatrix}=\begin{bmatrix} \frac{0.5}{h}v_{1(n)}+\sin(nh) \\ 0 \\ -\frac{0.1}{h}i_{3(n)} \end{bmatrix}$$

下面以欧拉算法为例，讨论离散时间结点方程的直接建立。

设方程为

$$(\boldsymbol{G}+\frac{1}{h}\boldsymbol{C})\boldsymbol{x}_{(n+1)}=\boldsymbol{B} \tag{7－25}$$

电阻、受控源、电压源、电流源在 $\boldsymbol{G}$ 和 $\boldsymbol{B}$ 中的填入已在第六章中作了介绍，此处不再重复。对电容元件，根据式(7－21)和图 7－19 所示的电流源复合支路，在电容连接结点处的 KCL 为

结点 p： $\frac{C}{h}\{v_{p(n+1)}-v_{n(n+1)}\}+\cdots=\frac{C}{h}v_{(n)}$

结点 n： $\frac{C}{h}\{v_{n(n+1)}-v_{p(n+1)}\}+\cdots=-\frac{C}{h}v_{(n)}$

若用 matc 表示矩阵 $\boldsymbol{C}$,vn 表示 $v_{(n)}$,由以上两式可写出以下编程语句:

```
matc(p,p) = matc(p,p) + C;
matc(n,n) = matc(n,n) + C;
matc(p,n) = matc(p,n) - C;
matc(n,p) = matc(n,p) - C;
ic = C/h * vn;
b(p) = b(p) + ic;
b(n) = b(n) - ic;
```

对电感元件,设电流变量为 i_m,为第 m 个方程变量,根据式(7-23)和图 7-22 所示的电压源复合支路,在电感连接结点处的 KCL 和电感支路的 VCR 分别为

结点 p: $i_{m(n+1)} + \cdots = 0$

结点 n: $-i_{m(n+1)} + \cdots = 0$

VCR: $v_{p(n+1)} - v_{n(n+1)} - \dfrac{L}{h} i_{m(n+1)} + \cdots = -\dfrac{L}{h} i_{m(n)}$

若用 matc 表示矩阵 $\boldsymbol{C}$,用 matg 表示矩阵 $\boldsymbol{G}$,用 imn 表示 $i_{m(n)}$,根据以上三式的编程语句为

```
matg(p,m) = matg(p,m) + 1;
matg(n,m) = matg(n,m) - 1;
matg(m,p) = matg(m,p) + 1;
matg(m,n) = matg(m,n) - 1;
matc(m,m) = matc(m,m) - L;
b(m) = b(m) - L/h * imn;
```

在编写时域分析程序时,对每一离散时间不必重新建立方程,矩阵 $\boldsymbol{G}$ 和 $\boldsymbol{C}$ 可事先建立好,在每一计算时间处,只需要修改与电容、电感、电压源、电流源有关的右端相量 $\boldsymbol{B}$。对图 7-23,方程为

$$\left(\underbrace{\begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{G} + \frac{1}{h}\underbrace{\begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -0.1 \end{bmatrix}}_{C}\right)\begin{bmatrix} v_{1(n+1)} \\ v_{2(n+1)} \\ i_{3(n+1)} \end{bmatrix} = \underbrace{\begin{bmatrix} \dfrac{0.5}{h} v_{1(n)} + \sin(nh) \\ 0 \\ -\dfrac{0.1}{h} i_{3(n)} \end{bmatrix}}_{B}$$

编写一个通用的时域分析程序,除要考虑积分方法的稳定性、精度、适用范围外,还要考虑以下几个因素。

(1) 每一时间处时间间隔的选取。由于电路中各固有频率的实部相差比较悬殊,在动态过程的起始阶段,由于响应的一些固有分量衰减较快,计算的时间间隔就必须取的比较小,否则将引起比较大的计算误差,使计算结果中遗漏了这些分量的存在。在动态过程到达一定阶段后,时间间隔的取值相对来说要比较大,以节省计算时间。总之,计算的时间间隔要取得适当,太小使得舍入误差增加,计算花费的时间较多,太大则不能反映电路的真实行为。如何合理的选取每一时间间隔,要根据所采用的积分方法和误差容限确定,这一过程应该由计算机自动完成。

(2) 对线性多步法,当时间间隔改变后,在此之前的值就不能直接利用,这时需要重新使用后向欧拉法起步,或采用变步长方法求解。

(3) 在使用积分算法时,原则上不能跨越信号及其导数的间断点计算,因此,程序应该对这些时间准确定位。研究表明,后向欧拉法和 Gear 法由于只使用当前时间处变量的导数值,而没有使用在此之前的导数值,因而能用于从间断点起始的计算。

第八章 频域分析

若线性电路的所有输入信号为相同频率的正弦信号，则电路微分方程的特解为正弦函数，其频率与输入信号的相同，而微分方程的齐次解与其特征根的指数函数有关。当特征根的实部小于零时（即电路是稳定的），对充分大的时间，齐次解几乎为零，故电路的全响应近似为正弦函数。如果设正弦输入信号在负无限大时刻施加于电路，则在任一时间处，电路已经经历了充分长的时间，电路的响应就为正弦信号，这种响应的分析称为正弦分析。

正弦分析的基本方法是把电路中的所有正弦信号变换成相量，对电路建立相量模型，依据相量模型与电阻电路的对比关系，列出电路的相量方程。由于相量是复数，电路元件的相量 VCR 为复数代数关系，因而电路的相量方程就为复数代数方程。在相量方程中，独立变量为频率，而不是时间，输出信号的相量为频率的函数，通常把这种对信号的时间变量变换成频率变量而后再通过逆变换求得时域信号的分析称为频域分析。频域分析的完整概念在学习了傅里叶变换内容后才能完全建立。

本章首先介绍复数方程的求解及其建立方法，其次介绍电路分析函数 sana 的正弦分析功能，通过一些典型例题说明它在正弦分析中的应用，内容包括基本电路的分析，含有磁耦合电感电路的分析，三相电路的分析等。在此之后，介绍电路频率响应的计算和电路的正弦叠加分析。最后，简要介绍电路的频域符号分析，其主要目的是为正在学习电路课程的读者提供一种分析工具，使手工计算电路函数、元件值等比较困难的问题变得简单化。

8.1 相量方程及其求解

对一个稳定的线性电路，如果所有输入信号为相同频率的正弦函数，电路的（稳态）响应也一定为与输入信号频率相同的正弦函数。电路正弦分析的基本方法是将正弦电压和电流变换为相量，依据电路的相量模型建立复数代数方程，设方程为

$$A\dot{X}=B \tag{8-1}$$

求解上式的 MATLAB 指令格式为

```
X = A\B
```

有关复数运算的函数如下：

```
abs(c)              %   求复数 c 的模
angle(c)            %   求复数 c 的相角,用弧度表示
angle(c) * 180/pi   %   求复数 c 的相角,用度表示
real(c)             %   求复数 c 的实部
imag(c)             %   求复数 c 的虚部
conj(c)             %   求复数 c 的共轭
```

在电路的正弦分析中,相量是复数,一般以指数形式给出,而且相角常常用度表示。然而,MATLAB 中复数以代数形式显示,为方便使用,读者可自编一些转换函数,例如求相角度数的函数:

```
function p = degree(c)
%DEGREE   Phase angle in degrees.
%    p = angle(c) * 180/pi;
%
p = angle(c) * 180/pi;
```

复数运算中经常遇到:(1)将复数的指数形式转化为代数形式;(2)将两个实数组合为一个复数;(3)屏幕显示复数的模和相角。下面给出的 cmplx 函数可完成这些功能。

```
function [c,d] = cmplx(m,d)
%CMPLX   m - - - magnitude;   d - - - phase in degree.
%    c = m. * exp(j * d * pi/180).
%    d = m + i * d;
% or
% [c,d] = cmplx(m)
% m - - - complex number; c - - - magnitude; d - - - phase in degree.
% Display  magnitude and phase angle in degree of complexes.
% Example:
% [c,d] = cmplx(5,53.13) yield c = 3 + 4i,d = 5 + 53.13i
% [m,d] = cmplx(3 + 4i) yield m = 5,d = 53.13
switch(nargin)
```

```
case nargin >2
     error('Too many input arguments')
case 2
      c = m. * exp(i * d * pi/180);
      d = m + i * d;
case 1
     [nr,nc] = size(m);
     vname = inputname(1);
     if isempty(vname)
        vname =' ans ';
     end
     c = abs(m);
     d = degree(m);
     fprintf('%s  =  \n',vname)
     m = [c,d];
     %   change column position for matrix m.
     for k =2:nc
          temp = m(:,k);
          n = nc + k - 1;
          m(:,k) = m(:,n);
          m(:,n) = temp;
     end
     disp(m);
end
```

例如在指令工作窗口键入

```
cmplx(5,53.13)
```

屏幕显示

```
ans  =  3 +4i
```

它把复数的指数形式转化为代数形式。

若键入

```
[c1,c2] = cmplx(5,53.13)
```

屏幕显示

```
c1  =  3 +4i, c2  =  5 +53.13i
```

其中,c1 把复数的指数形式转换为代数形式,c2 将函数中的两个输入数据组合为复数。

再如:

```
cmplx(3 + j * 4);
```

将显示复数的模和相角的度数,输出为

```
ans =      5          53.13
```

若键入

```
[m,d] = cmplx(3 + j * 4)
```

除显示复数的模和相角的度数外,还给出

```
m = 5
d = 53.13
```

例 8-1　正弦分析的基本方法

图 8-1 所示电路中,$R_1 = 10\ \Omega$,$L = 0.5\ \mathrm{H}$,$R_2 = 1\ 000\ \Omega$,$C = 10\ \mu\mathrm{F}$,$\dot{V} = 100\angle 0°\ \mathrm{V}$,$\omega = 314\ \mathrm{rad/s}$。求支路电流和结点电压。

解　先建立有关数学方程。令

$$Y_1 = \frac{1}{R_1 + \mathrm{j}\omega L}$$

运用结点法得结点 1 的电压

$$\dot{V}_1 = \frac{Y_1}{Y_1 + \mathrm{j}\omega C + \dfrac{1}{R_2}}\dot{V}$$

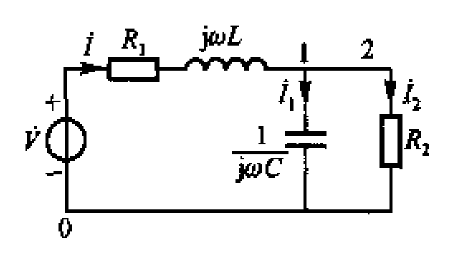


图 8-1　例 8-1 电路

则支路电流

$$\dot{I} = Y_1(\dot{V} - \dot{V}_1)$$

$$\dot{I}_1 = \mathrm{j}\omega C\,\dot{V}_1$$

$$\dot{I}_2 = \frac{1}{R_2}\dot{V}_1$$

依据以上公式编写的 M 文件如下:

```
r1 = 10; l = 0.5; r2 = 1000; c = 10e-6; v = 100; w = 314;      % ---(1)
y1 = 1/(r1 + j * w * l);
v1 = y1 * v/(y1 + j * w * c + 1/r2);
i0 = y1 * (v - v1);                                              % ---(2)
i1 = j * w * c * v1; i2 = v1/r2;
cmplx(v1); cmplx(i0); cmplx(i1); cmplx(i2);                      % ---(3)
```

其中，语句(1)给出元件的值，角频率用 w 表示；语句(2)用 i0 表示电流 $\dot{I}$，注意勿用 i 表示，因 i 在 MATLAB 中已作为复数的虚单位；语句(3)显示电压、电流的振幅和相角，相角用度表示。结果如下：

```
v1 =   181.73      -20.022
i0 =   0.59886      52.313
i1 =   0.57062      69.978
i2 =   0.18173     -20.022
```

即
$$\dot{V}_1 = 181.73 \angle -20.022^\circ \ \text{V}$$

如果用网孔法求解该题，取 $\dot{I}$ 和 $\dot{I}_2$ 为变量，方程为

$\dot{I}$ 网孔：
$$\left(R_1 + \text{j}\omega L + \frac{1}{\text{j}\omega C}\right)\dot{I} - \frac{1}{\text{j}\omega C}\dot{I}_2 = \dot{V}$$

$\dot{I}_2$ 网孔：
$$-\frac{1}{\text{j}\omega C}\dot{I} + \left(R_2 + \frac{1}{\text{j}\omega C}\right)\dot{I}_2 = 0$$

用网孔电流求解电流 $\dot{I}_1$ 和结点电压 $\dot{V}_1$ 的公式为

$$\dot{I}_1 = \dot{I} - \dot{I}_2$$
$$\dot{V}_1 = \frac{1}{\text{j}\omega C}\dot{I}_1$$

根据以上 4 个公式编写的 M 文件如下：

```
r1 = 10; l = 0.5; r2 = 1000; c = 10e-6; v = 100; w = 314;
a = [r1 + j*w*l + 1/(j*w*c), -1/(j*w*c); -1/(j*w*c), r2 + 1/
    (j*w*c)];
b = [v;0];
x = a\b;
i0 = x(1); i2 = x(2);
i1 = i0 - i2;
v1 = i1/(j*w*c);
cmplx(v1); cmplx(i0); cmplx(i1); cmplx(i2);
```

对复杂电路，一般要采用结点法求解。在第六章已经介绍了电阻电路方程的建立方法，当电路做正弦分析时，依据相量模型与电阻电路的对比关系，相量结点方程的建立也非常容易。但是，在 $\omega = 0$ 时由于电感的阻抗为零，也即导纳为无限大，若按导纳填入方程则会引起困难。为了避免该情况，使方程也能用于直流分析，故常常把电感电流也作为方程的变量。此外，这样做还主要是由于能够容易处

理磁耦合电感。当把电感的电流也作为方程变量后,电感连接结点的 KCL 方程就不需要其电压电流关系式,该关系式作为结点方程的补充方程给出。

按上述方法,相量结点方程的一般形式为

$$(\boldsymbol{G}+\mathrm{j}\omega\boldsymbol{C})\dot{\boldsymbol{X}}=\dot{\boldsymbol{B}} \tag{8-2}$$

其中,$\dot{\boldsymbol{X}}$ 是由结点电压和一些电流构成的列向量;电阻和受控源的值在矩阵 $\boldsymbol{G}$ 中填入;电容和电感的值在矩阵 $\boldsymbol{C}$ 中填入,$\dot{\boldsymbol{B}}$ 与输入有关;ω 为正弦角频率。

以图 8-2 所示电路为例,取结点电压 $\dot{V}_1$、$\dot{V}_2$ 和电感电流 $\dot{I}_L$ 为变量,有

结点 1: $\frac{1}{R_1}\dot{V}_1+\frac{1}{R_2}(\dot{V}_1-\dot{V}_2)+\dot{I}_L=\dot{I}_S$

结点 2: $\frac{1}{R_2}(\dot{V}_2-\dot{V}_1)+\frac{1}{R_3}\dot{V}_2+\mathrm{j}\omega C\dot{V}_2-\dot{I}_L=0$

电感: $\dot{V}_1-\dot{V}_2-\mathrm{j}\omega L\dot{I}_L=0$

将以上方程整理成矩阵,有

$$\left[\underbrace{\begin{bmatrix}\frac{1}{R_1}+\frac{1}{R_2} & -\frac{1}{R_2} & 1\\ -\frac{1}{R_2} & \frac{1}{R_2}+\frac{1}{R_3} & -1\\ 1 & -1 & 0\end{bmatrix}}_{\boldsymbol{G}}+\mathrm{j}\omega\underbrace{\begin{bmatrix}0 & 0 & 0\\ 0 & C & 0\\ 0 & 0 & -L\end{bmatrix}}_{\boldsymbol{C}}\right]\begin{bmatrix}\dot{V}_1\\ \dot{V}_2\\ \dot{I}_L\end{bmatrix}=\underbrace{\begin{bmatrix}\dot{I}_S\\ 0\\ 0\end{bmatrix}}_{\boldsymbol{B}}$$

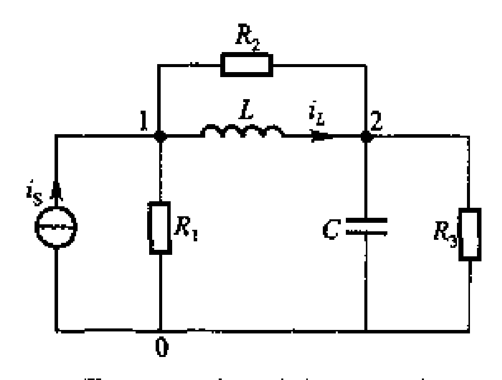


图 8-2 建立结点方程示例

下面考虑动态元件在方程中的填入问题。

(1) 电容。

参看图 8-3,对结点 p 和结点 n 应用 KCL,可得

结点 p: $\mathrm{j}\omega C(\dot{V}_p-\dot{V}_n)+\cdots=0$

结点 n: $\mathrm{j}\omega C(\dot{V}_n-\dot{V}_p)+\cdots=0$

则电容只在矩阵 $\boldsymbol{C}$ 中填入，若在程序中用 matc 表示式(8-2)中的矩阵 $\boldsymbol{C}$，用 C 表示电容值 C，程序如下：

```
matc(p,p) = matc(p,p) + C;
matc(n,n) = matc(n,n) + C;
matc(p,n) = matc(p,n) - C;
matc(n,p) = matc(n,p) - C;
```

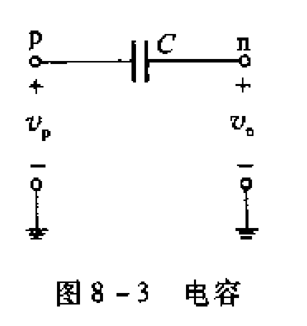


图 8-3 电容

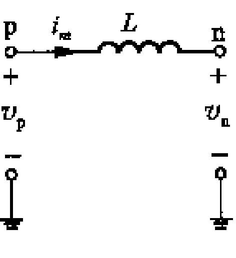


图 8-4 电感

(2) 电感。

参看图 8-4，对结点 p 和结点 n 应用 KCL，可得

结点 p： $\dot{I}_m + \cdots = 0$

结点 n： $-\dot{I}_m + \cdots = 0$

电感的电压电流关系为

$$\dot{V}_p - \dot{V}_n - j\omega L\,\dot{I}_m = 0$$

设电感电流 $\dot{I}_m$ 为第 m 个方程变量，若在程序中用 matg 表示矩阵 $\boldsymbol{G}$，与以上公式对应的语句如下：

```
matg(p,m) = matg(p,m) + 1;
matg(n,m) = matg(n,m) - 1;
matg(m,p) = matg(m,p) + 1;
matg(m,n) = matg(m,n) - 1;
matc(m,m) = matc(m,m) - L;
```

(3) 互感。

参看图 8-5，结点的 KCL 方程为

结点 p_a： $\dot{I}_a + \cdots = 0$

结点 n_a： $-\dot{I}_a + \cdots = 0$

结点 p_b： $\dot{I}_b + \cdots = 0$

结点 n_b：　$-\dot{I}_b+\cdots=0$

磁耦合电感的电压电流关系式为

L_a 支路：　$\dot{V}_{p_a}-\dot{V}_{n_a}-j\omega(L_a\dot{I}_a+M\dot{I}_b)=0$

L_b 支路：　$\dot{V}_{p_b}-\dot{V}_{n_b}-j\omega(M\dot{I}_a+L_b\dot{I}_b)=0$

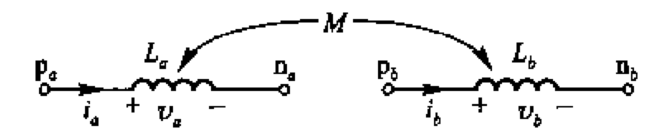


图 8－5　磁耦合电感

设电感电流 $\dot{I}_a$、$\dot{I}_b$ 分别为第 a 个和第 b 个方程变量，自感值在方程中的填入已在(2)中给出，而互感值的填入如下：

```
matc(a,b) = matc(a,b) - M;
matc(b,a) = matc(b,a) - M;
```

8.2 电路的正弦分析

手工建立电路方程容易出错，不适合复杂电路。本节介绍函数 sana 的正弦数值分析功能，该函数建立式(8－2)所示的结点方程，电阻、受控源的输入格式与第六章中的相同，动态元件的描述格式如下：

电容元件：　'c＊＊＊　正端　负端　电容值'

电感元件：　'l＊＊＊　正端　负端　电感值'

互感：　'm＊＊＊　电感名　电感名　互感值'

注意：由于程序把电感电流作为方程变量，因此电感名不得相同。在显示的结果中，电感电流用字母 i 后紧跟电感名表示，如 il3 表示电感 l3 的电流，电流方向从其正端指向负端。当电感间由磁耦合关系时，每两个电感的正端为对应端时，互感取正值，否则取负值。

电压源和电流源的值为复数，对指数形式，可采用作者提供的函数 cmplx 对数据进行转换。

函数 sana 作正弦分析时的调用格式为

```
sana(ckt,w)
[X,B,G,C] = sana(ckt,w)
```

其中，ckt 为描述电路的群数组；w 为电源的角频率，函数的输出变元为式(8－2)

中的有关矩阵。

例 8－2　正弦分析示例

图 8－6 所示 *RLC* 串联电路，$v(t)=100\sqrt{2}\cos(5\,000t+30^\circ)$ V，$R=15\ \Omega$，$L=12$ mH，$C=5\ \mu$F，求电容电压和电感电流，并计算电源发出的复功率。

解　M 文件如下：

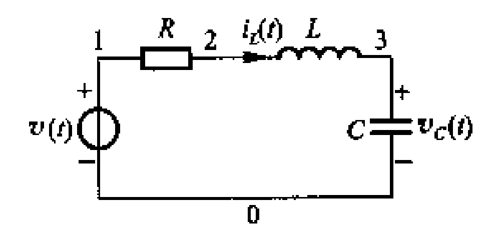


图 8－6　*RLC* 串联电路

```
ckt = {'v 1 0 cmplx(100,30)'
       'r 1 2 15'
       'l 2 3 12e-3'
       'c 3 0 5e-6'};
w = 5000;
x = sana(ckt,w);
cmplx(x);
vc = v3;
cmplx(vc);cmplx(il);
s = -v1 * conj(iv)
```

程序的最后一条语句中，由于电压源的电压和电流取关联方向，故电压源发出的复功率等于电压乘电流共轭复数的负值。

屏幕显示如下：

```
Nodal voltages v* * and element currents i* * :
v1    v2    v3       iv        il
x  =      100           30
           80        66.87
          160      -113.13
            4       156.87
            4       -23.13
vc =      160      -113.13
il =        4       -23.13
s  =      240 +       320i
```

即

$$\dot{V}_C=160\,\underline{/-113.13^\circ}\ \text{V}$$

$$\dot{I}_L=4\,\underline{/-23.13^\circ}\ \text{A}$$

$$\overline{S}=240+\text{j}320\ \text{V}\cdot\text{A}$$

当阻抗给定，而未给出频率时，可假设角频率 $\omega = 1$ rad/s，对各阻抗用与其等效的元件进行输入。另一种方法是把阻抗按电阻的形式输入，只不过值是复数。

例 8－3　阻抗的输入格式及等效阻抗的求解

电路如图 8－7 所示，已知 $Z_1 = 5\angle{-45^\circ}\ \Omega$，$Z_2 = 1 + \mathrm{j}2\ \Omega$，$Z_3 = 3 - \mathrm{j}4\ \Omega$，求等效阻抗和等效导纳。

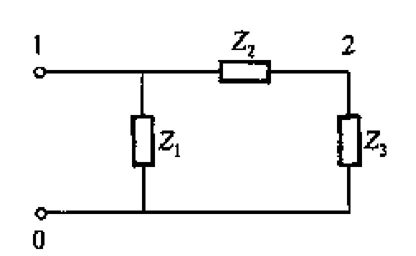
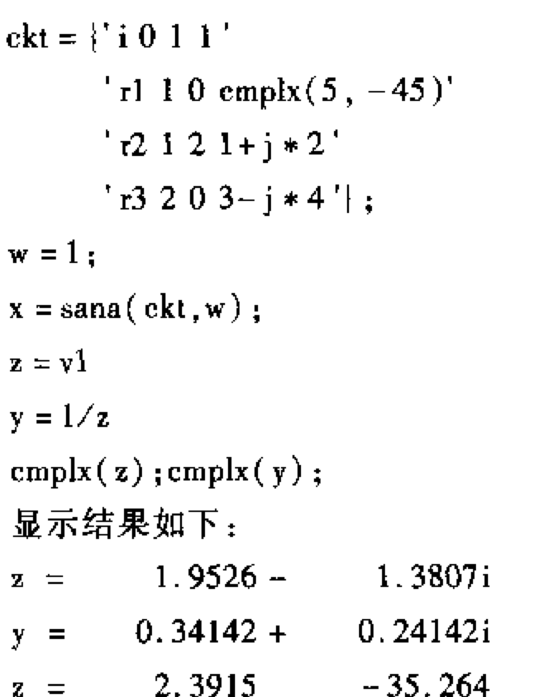


图 8－7　例 8－3 电路

解　若给端口施加 1 A 的电流源，则端口电压在数值上等于等效阻抗。M 文件如下：



```
ckt = {'i 0 1 1'
       'r1 1 0 cmplx(5, -45)'
       'r2 1 2 1+j*2'
       'r3 2 0 3-j*4'};
w = 1;
x = sana(ckt,w);
z = v1
y = 1/z
cmplx(z);cmplx(y);
```

显示结果如下：

```
z  =     1.9526 -      1.3807i
y  =    0.34142 +     0.24142i
z  =     2.3915       -35.264
y  =    0.41815        35.264
```

其中，后两行数据为阻抗、导纳的模值和相角。

例 8－4　三相电路

图 8－8 所示三相电路，设对称三相电源的线电压为 380 V，电阻 $R = 10\ \Omega$，阻抗 $Z_1 = \mathrm{j}4\ \Omega$，$Z_2 = 8 - \mathrm{j}4\ \Omega$，求流出电源的线电流。

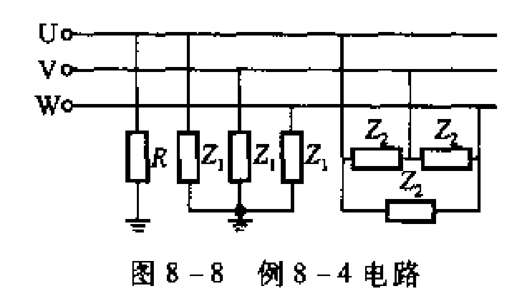


图 8－8　例 8－4 电路

解　三相电源有星形和三角形两种连接方式，对理想三角形联结电压源，由于存在电压源回路，则电流的解不唯一。本题按星形联结电源处理。若用结点

1、2、3 分别表示 U、V、W，则 M 文件如下：

```
ckt = {'va 1 0 380/sqrt(3)'
       'vb 2 0 cmplx(380/sqrt(3), -120)'
       'vc 3 0 cmplx(380/sqrt(3),120)'
       'r 1 0 10'
       'l1a 1 0 4'
       'l1b 2 0 4'
       'l1c 3 0 4'
       'r2 1 2 8-j*4'
       'r2 2 3 8-j*4'
       'r2 3 1 8-j*4'};
w = 1;
x = sana(ckt,w);
ia = -iva; ib = -ivb; ic = -ivc;
cmplx(ia);cmplx(ib);cmplx(ic);
```

线电流的有效值和相角如下：

```
ia =    90.458      -14.036
ib =    69.378      -138.43
ic =    69.378      101.57
```

例 8-5 磁耦合电感

图 8-9 电路中，$L_1 = 1$ H，$L_2 = 2$ H，，$L_3 = 3$ H，$M_{12} = M_{13} = M_{23} = 1$ H，$R = 100\ \Omega$，$C = 0.1\ \mu F$，$v(t) = 220\sqrt{2}\cos(2\pi ft + 30°)$ V，$f = 50$ Hz，求电感电流。

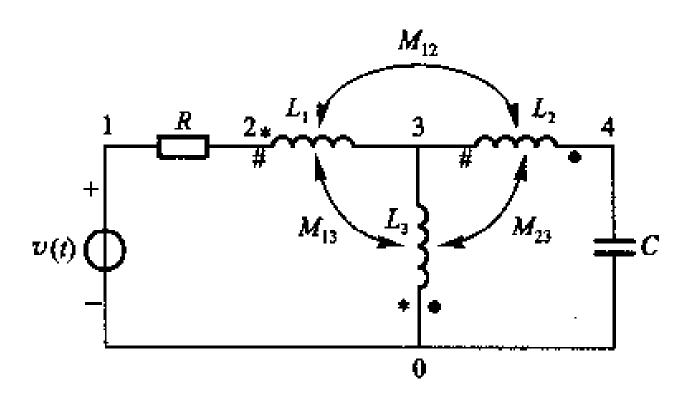


图 8-9 例 8-5 电路

解 M 文件如下：

```
ckt = {'v 1 0 cmplx(220,30)'
```

```
        'r 1 2 100'
        'l1 2 3 1'
        'l2 4 3 2'
        'l3 0 3 3'
        'm12 l1 l2 -1'
        'm23 l2 l3 1'
        'm13 l1 l3 1'
        'c 4 0 0.1e-6'}
f=50; w=2*pi*f;
x=sana(ckt,w);
cmplx(il1);cmplx(il2);cmplx(il3);
```

电流的有效值和相角如下：

```
il1 =       0.34579     -50.957
il2 =    3.49e-017     -170.96
il3 =       0.34579     129.04
```

利用互感消去法可得电流 $\dot{I}_{L_2}=0$，由于舍入误差的影响，程序计算出的 $\dot{I}_{L_2}$ 为一个非常小的数。

例 8-6 理想变压器

图 8-10(a)所示电路含有理想变压器，设 $\dot{V}=1$ V，编写求电压 $\dot{V}_2$ 的程序。

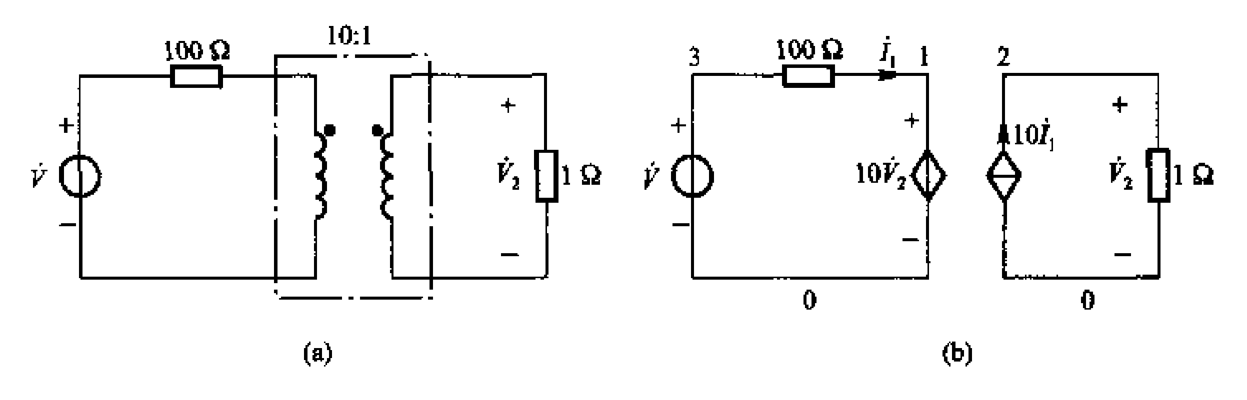


图 8-10 例 8-6 电路

解 用受控源表示理想变压器，电路如图 8-10(b)所示。由于该电路是非连通的，在使用结点法时，必须对各连通部分都指定一个参考结点。M 文件如下：

```
ckt={'v 3 0 1'
        'r1 3 1 100'
```

```
        'e 1 0 2 0 10'
        'g 0 2 e 10'
        'r2 2 0 1'};
w = 1;
x = sana(ckt,w);
v2
```

8.3 频率响应

设电路是稳定的，只有一个输入，输入信号 $x(t)$ 是角频率为 ω 的正弦信号，即

$$x(t)=X\cos(\omega t+\varphi_x)$$

令 $\dot{X}=Xe^{j\varphi_x}$，用相量法求解电路，设输出信号的相量 $\dot{Y}=Ye^{j\varphi_y}$，于是

$$y(t)=Y\cos(\omega t+\varphi_y)$$

根据电路相量模型的齐次性，输出信号的相量 $\dot{Y}$ 与输入信号的相量 $\dot{X}$ 成正比关系，其比值与 $\dot{X}$ 无关，而是 $j\omega$ 的函数，如果把 $j\omega$ 作为参量，$\dot{Y}$ 与 $\dot{X}$ 之比定义为 $y(t)$ 的（频域）电路函数，设用 $H(j\omega)$ 表示，则

$$H(j\omega)=\frac{\dot{Y}}{\dot{X}} \tag{8-3}$$

电路函数以往也称为网络函数，如果视电路为一种系统，$H(j\omega)$ 也称为系统函数。$H(j\omega)$ 一般为复数，可表示为

$$H(j\omega)=|H|e^{j\angle H} \tag{8-4}$$

其中，$|H|$ 称为幅度函数；$\angle H$ 称为相位函数。根据电路函数的定义，输出相量

$$\dot{Y}=H(j\omega)\dot{X}=|H|Xe^{j\varphi_x+\angle H}=Ye^{j\varphi_y}$$

或

$$\begin{cases}Y=|H|X\\ \varphi_y=\angle H+\varphi_x\end{cases}$$

即输出信号的振幅等于电路函数的幅度与输入信号振幅的乘积，而输出信号的相角等于电路函数的相角与输入信号的相角之和。

当电路的输入信号由多个不同频率的正弦分量组成时，如

$$\begin{aligned}x(t)&=X_1\cos(\omega_1 t+\varphi_{x1})+X_2\cos(\omega_2 t+\varphi_{x2})+X_3\cos(\omega_3 t+\varphi_{x3})\\&=\mathrm{Re}[\dot{X}_1e^{j\omega_1 t}+\dot{X}_2e^{j\omega_2 t}+\dot{X}_3e^{j\omega_3 t}]\end{aligned}$$

其中，

$$\dot{X}_k = X_k e^{j\varphi_{xk}} \qquad (k=1,2,3)$$

根据电路函数的定义，各正弦输入分量对应的输出分量应该为

$$\dot{Y}_k = Y_k e^{j\varphi_{yk}} = H(j\omega_k)\dot{X}_k \qquad (k=1,2,3)$$

则

$$\begin{aligned} y(t) &= \mathrm{Re}[\dot{Y}_1 e^{j\omega_1 t} + \dot{Y}_2 e^{j\omega_2 t} + \dot{Y}_3 e^{j\omega_3 t}] \\ &= Y_1\cos(\omega_1 t+\varphi_{y1}) + Y_2\cos(\omega_2 t+\varphi_{y2}) + Y_3\cos(\omega_3 t+\varphi_{y3}) \end{aligned}$$

由此可看出，电路函数反映了电路在不同频率处对正弦输入信号的放大和移位的程度，以振幅为例，当$|H(j\omega_1)| > |H(j\omega_2)|$时，说明电路对频率为$\omega_1$的正弦信号的增益要高于频率为$\omega_2$的正弦信号，为了观察电路在不同频率处增益的总体状况和相对大小，$|H(j\omega)|$随频率ω的变化曲线称为电路的幅频响应。同理，相位函数$\underline{/\ H(j\omega)}$随频率$\omega$的变化曲线称为电路的相频响应。

电路函数与输入输出微分方程也有着密切联系。如给定

$$a_0\frac{d^2y(t)}{dt^2} + a_1\frac{dy(t)}{dt} + a_2 y(t) = b_0\frac{d^2x(t)}{dt^2} + b_1\frac{dx(t)}{dt} + b_2 x(t)$$

对正弦信号$x(t)$，则输出、输入相量之间的关系为

$$[a_0(j\omega)^2 + a_1(j\omega) + a_2]\dot{Y} = [b_0(j\omega)^2 + b_1(j\omega) + b_2]\dot{X}$$

于是电路函数$H(j\omega)$就为

$$H(j\omega) = \frac{\dot{Y}}{\dot{X}} = \frac{b_0(j\omega)^2 + b_1(j\omega) + b_2}{a_0(j\omega)^2 + a_1(j\omega) + a_2} = \frac{B(j\omega)}{A(j\omega)}$$

由此可见，电路函数可表示为两个$j\omega$多项式之比形式。

当电路函数的表达式给定时，对每一频率，可用 MATLAB 的多项式求值函数 polyval 计算分子、分母多项式的值。

当电路给定时，用计算机绘制频率响应曲线的方法非常简单，在给定频率范围内，取足够数量的离散频率，假定输入信号的频域值为 1，求出输出信号的振幅和相位，则输出信号振幅和相位随频率的变化关系就是电路的频率响应。

例 8-7 电路函数的频率响应曲线

图 8-11 所示 RLC 串联电路，已知 $R=1\ \Omega, L=2\ \mathrm{H}, C=0.5\ \mathrm{F}$，求电流 $i(t)$ 的电路函数，并在 $0.02\ \mathrm{rad/s} \leqslant \omega \leqslant 3\ \mathrm{rad/s}$ 的频率范围内绘制幅频响应和相频响应曲线。

解 设正弦角频率为ω，则输入阻抗

$$Z(\mathrm{j}\omega) = R + \mathrm{j}\omega L + \frac{1}{\mathrm{j}\omega C}$$

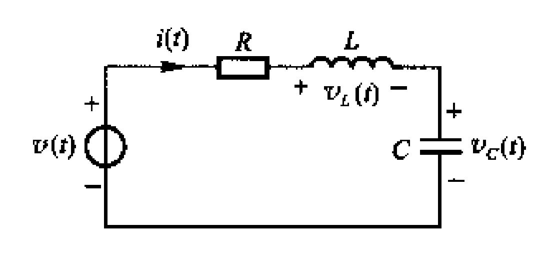


图 8 - 11 *RLC* 串联电路

电路函数

$$H(\mathrm{j}\omega) = \frac{\dot{I}}{\dot{V}} = \frac{1}{Z(\mathrm{j}\omega)}$$

$$= \frac{1}{R + \mathrm{j}\omega L + \frac{1}{\mathrm{j}\omega C}}$$

$$= \frac{\mathrm{j}\omega C}{(\mathrm{j}\omega)^2 LC + \mathrm{j}\omega RC + 1}$$

代入已知数据得

$$H(\mathrm{j}\omega) = \frac{0.5(\mathrm{j}\omega)}{(\mathrm{j}\omega)^2 + 0.5(\mathrm{j}\omega) + 1}$$

计算频率响应的 M 文件如下：

```
w = [0.02:0.02:3];
s = j * w;
b = [0.5,0];        % coefficients of the numerator polynomial
a = [1,0.5,1];      % coefficients of the denominator polynomial
h = polyval(b,s)./polyval(a,s);
subplot(1,2,1)
plot(w,abs(h))
subplot(1,2,2)
plot(w,degree(h))
```

频率响应曲线如图 8 - 12 所示，其中电路函数的相位以度为单位。从幅频响应曲线可看出，在 $\omega = 1$ rad/s 处电路函数的振幅最大，在偏离 $\omega = 1$ rad/s 较远处振幅相对较小，因此，该电路当以电流为输出时，它具有选取输入信号中 $\omega = 1$ rad/s附近信号分量的功能。

频率响应曲线也可使用 MATLAB 中提供的函数绘制，该函数为

```
h = freqs(b,a,w)
```

其中，b 和 a 分别为分子、分母多项式的系数向量，注意多项式要以 jω 的降幂形式排列；w 为角频率向量；h 为电路函数的值。曲线的频率坐标采用对数刻度，幅度和相角分别用分贝、度表示。当然，在图形窗口中，用户也可对其修改，如可将频率坐标改为线性刻度。对例 8 - 7，使用 freqs 函数绘制频率响应曲线的 M

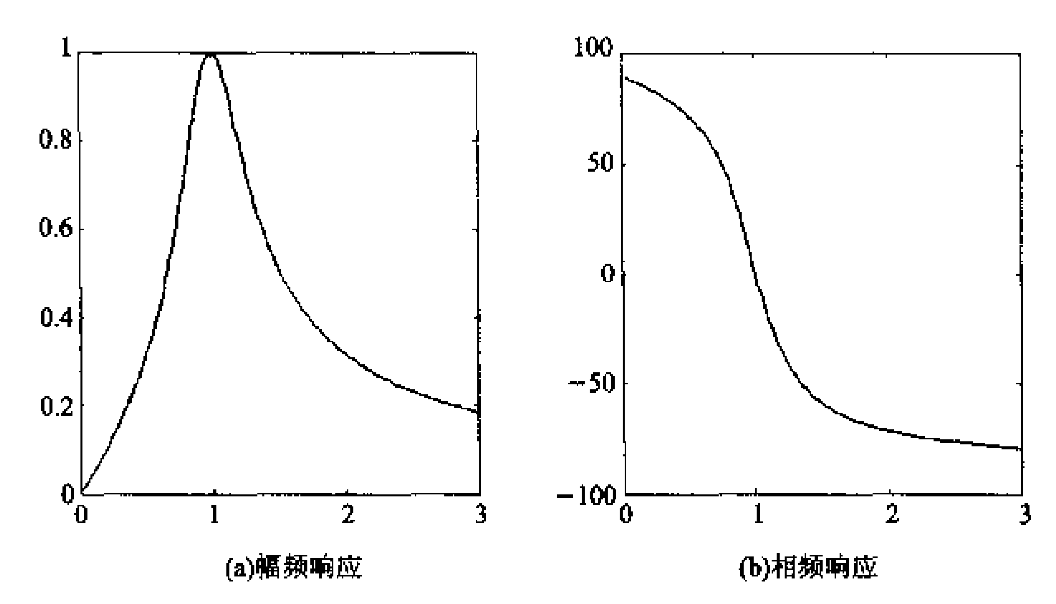


图 8－12　频率响应曲线

文件如下：

```
w = [0.02:0.02:3];
b = [0.5,0];
a = [1,0.5,1];
freqs(b,a,w)
```

有关曲线如图 8－13 所示。

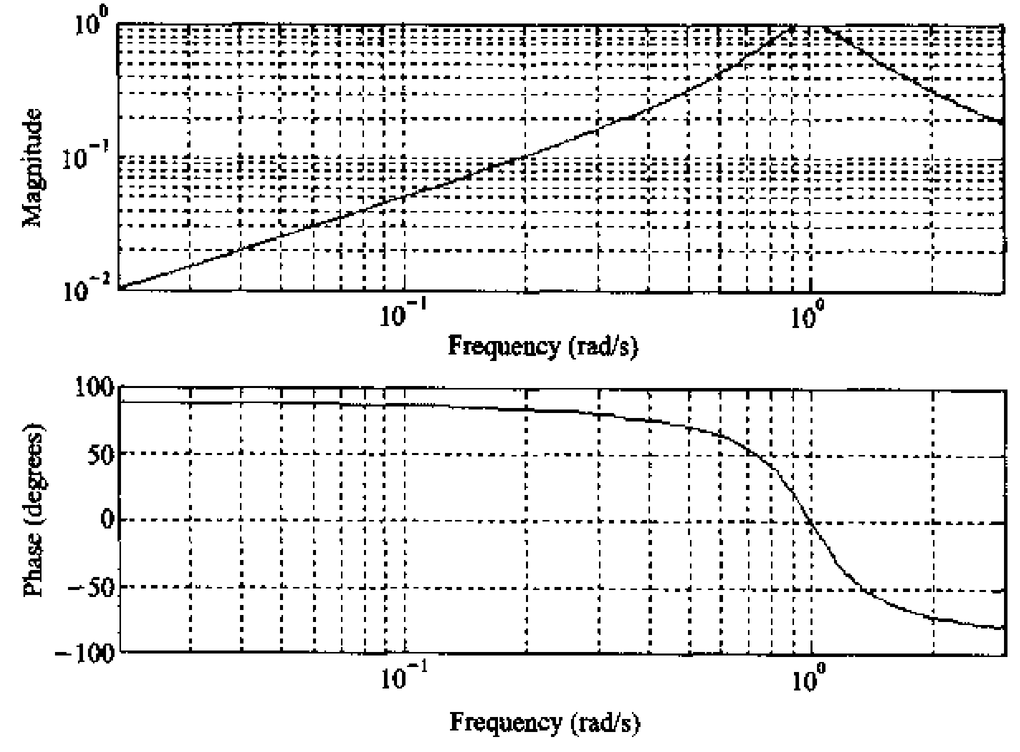


图 8－13　频率采用对数刻度的频率响应曲线

电路的频率响应也可用函数 sana 求解，格式为

sana(ckt,w)

其中，ckt 为描述电路的群数组；w 为角频率向量。利用 sana 函数的计算结果，再用 plot 函数绘制频率响应曲线。

例 8-8 电路的频率响应

绘制例 8-7 中电容电压和电感电流的幅频响应曲线。

解 M 文件如下：

```
w = [0.02:0.02:3];
ckt = {'v 1 0 1'
       'r 1 2 1'
       'l 2 3 2'
       'c 3 0 0.5'}
sana(ckt,w);
vl = v2 - v3; vc = v3;
plot(w,abs(vl),w,abs(vc))
grid
```

幅频响应曲线如图 8-14 所示。

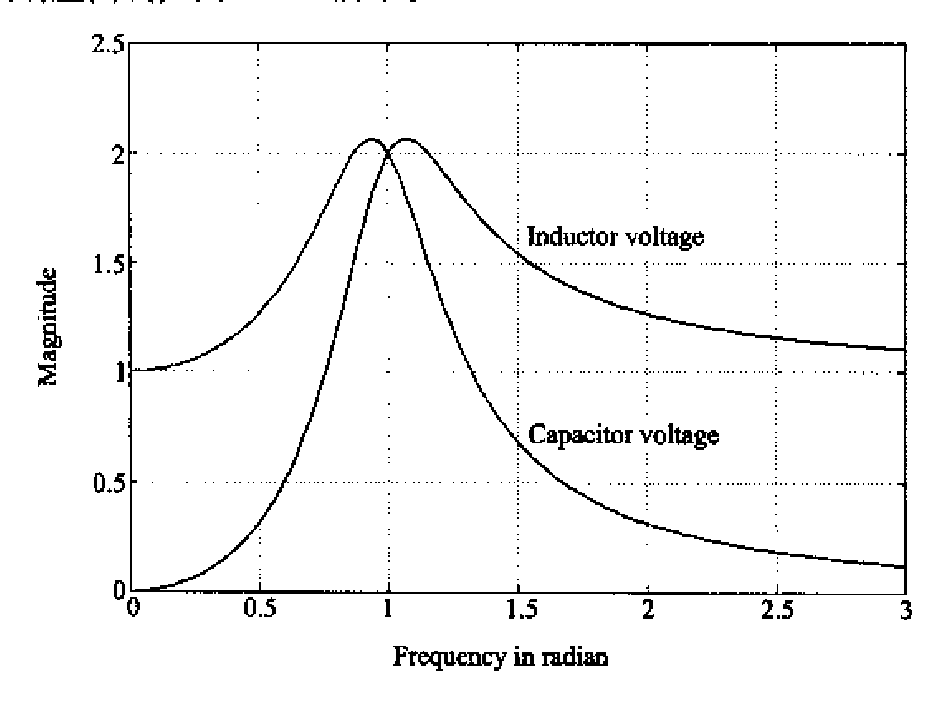


图 8-14 幅频响应

当电路的输入由多个不同频率的正弦信号组成时，用 sana 函数求解输出信

号也很方便,方法是:令输入为 1,求出电路在给定频率处电路函数的值,然后,根据给定输入信号写出由各相量组成的数组,再利用点乘运算求出各频率处的输出相量,根据该结果可写出输出表达式。

例 8-9 电路的正弦叠加分析

设图 8-11 所示电路的输入为

$$v(t) = 4\cos(0.5t + 30^\circ) + 3\cos t + 2\cos(1.5t + 15^\circ)\ \text{V}$$

求 $i(t)$ 和 $v_C(t)$。

解 输入信号的角频率数组和相量数组为

```
w = [0.5,1,1.5]
v = [cmplx(4,30),3, cmplx(2,15)]
```

M 文件如下:

```
w = [0.5,1,1.5];
ckt = {'v 1 0 1'
      'r 1 2 1'
      'l 2 3 2'
      'c 3 0 0.5'};
sana(ckt,w);
v = [cmplx(4,30), 3, cmplx(2,15)];
vc = v3;
disp('Values of the circuit functions')
vc = vc';il = il';v = v';   % Change to column vector
cmplx(vc); cmplx(il);
vc = vc. * v; il = il. * v;
disp('Phasors of the outputs')
cmplx(vc); cmplx(il);
```

输出结果为

```
vc =   1.2649      18.435
            2          90
      0.68599      149.04
il =  0.31623     -71.565
            1           0
       0.5145      59.036
Phasors of the outputs
```

```
vc =    5.0596      -11.565
             6           90
         1.372       134.04
il =    1.2649      -101.57
             3            0
         1.029       44.036
```

即

$$v_C(t) = 5.0596\cos(0.5t - 11.565°) + 6\cos(t + 90°) + 1.372\cos(1.5t + 134.04°)\ \text{V}$$

$$i(t) = 1.2649\cos(0.5t - 101.57°) + 3\cos t + 1.029\cos(1.5t + 44.036°)\ \text{A}$$

8.4 频域符号分析

在电路设计中,当电路结构给定时,要根据输入输出关系确定元件值,这时就需要使用符号推理的方法求出电路的解析解。本节通过几个例题说明 sana 的频域符号分析功能及其应用。

借助 sana 电阻电路的符号分析功能,若按电阻元件格式输入频域阻抗,输入电源指定为 1,函数 sana 就能求得电路函数。

例 8-10　频域电路函数的符号分析

若用符号表示图 8-11 所示电路中的元件值,求电容电压的电路函数。

解　令输入电压为 1,则这时的电容电压就是其电路函数。M 文件如下:

```
ckt = {'v 1 0 1'
       'r 1 2 R'
       'rl 2 3 j * w * L'
       '% Inductor'
       'rc 3 0 1/(j * w * C)'
       '% Capacitor'}
sana(ckt);
h = v3
```

结果为

```
    h = 1/(1 - w^2 * C * L + i * R * w * C)
```

即

$$H(j\omega)=\frac{1}{1-\omega^2CL+j\omega RC}$$

电路函数也可采用拉普拉斯变化法求解，这时描述电容和电感的数据格式较为简单，有关内容见第十一章内容。

例 8－11 元件值可调电路的正弦分析

图 8－15 所示电路为考虑电感线圈电阻的并联谐振电路，设电容 C 可调，正弦电源的角频率 $\omega=314$ rad/s。(1)绘出输入导纳角随 C 变化的曲线；(2)求输入导纳虚部为零时的电容 C。

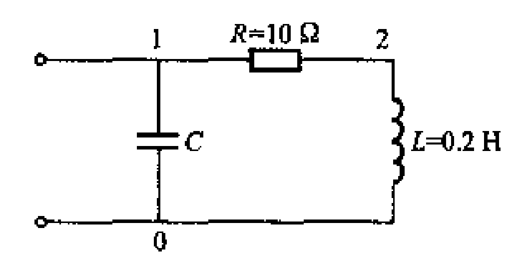


图 8－15 并联谐振电路

解 若对该题作近似估算，Q 值为

$$Q=\frac{\omega L}{R}=\frac{314\times0.2}{10}=6.28$$

由于 Q 值较大，故输入导纳虚部为零时有以下近似关系

$$\omega\approx\frac{1}{\sqrt{LC}}$$

或

$$C\approx\frac{1}{\omega^2L}=50.7\ \mu\text{F}$$

设电容在 10～100 μF 范围变化。M 文件如下：

```
ckt = {'v 1 0 1'
       'rc 1 0 1/(j*314*c)'
       'r 1 2 10'
       'rl 2 0 j*314*0.2'}
sana(ckt);
y = -iv;    % y - input admittance
c = 1e-6*(10:1:100);
y = subs(y,c,'c');   % substitute capacitance with numeric array
```

```
y = degree(y);          % determine the degree
plot(c,y),grid,
y = -iv;
y = imag(y);            % find the imaginary part of admittance
c = solve(y)            % solve the capacitance
y = subs(y,c,'c');
```

导纳角随电容的变化曲线如图 8-16 所示。当导纳的虚部为零时,程序求出

$$C = 49.458\ \mu\text{F}$$

可看出,估算值与它非常接近。

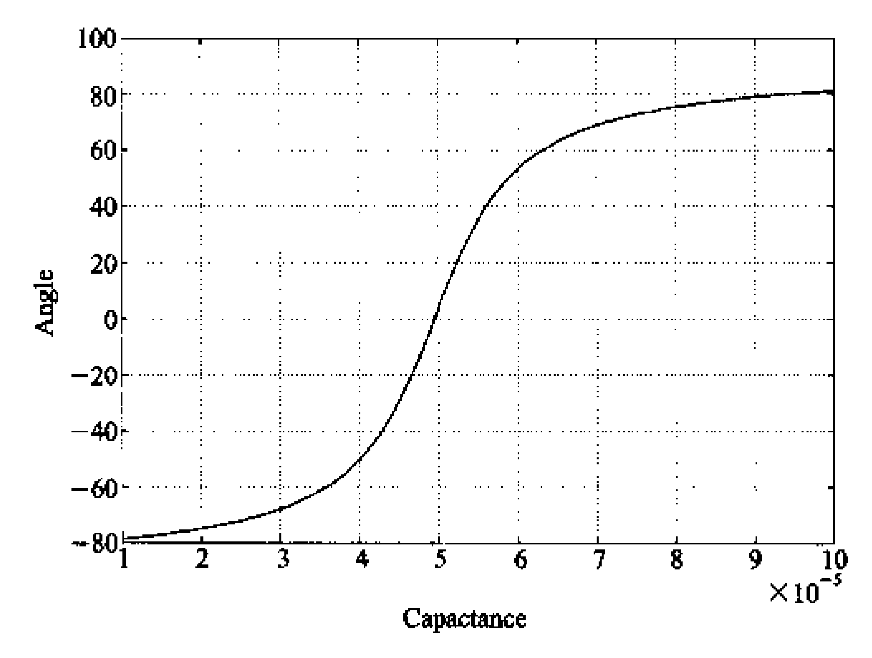


图 8-16 导纳角随电容的变化曲线

例 8-12 正弦最大功率

图 8-17 所示电路中,负载电阻 R 可调,求它为何值时可获得最大功率。

解 设电压源电压的有效值为 V,则电阻 R 吸收的功率为

$$P = R\frac{V^2}{(R+R_1)^2+X_1^2}$$

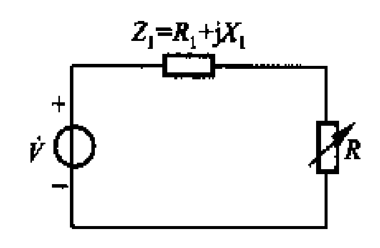


图 8-17 例 8-12 电路

当 R 获得最大功率时,应有 $\frac{\mathrm{d}P}{\mathrm{d}R}=0$。M 文件如下:

```
syms v r1 x1 r real
p = r * v^2/((r + r1)^2 + x1^2);
eq = diff(p,'r');          % dp/dr
```

```
rmax = solve(eq,'r')        % solve r
pmax = subs(p,rmax,'r')  % solve maximum power
```

输出结果为

```
rmax =
[   (r1^2 + x1^2)^(1/2)]
[ -(r1^2 + x1^2)^(1/2)]
pmax =
[    (r1^2 + x1^2)^(1/2) * v^2/((((r1^2 + x1^2)^(1/2) + r1)^2 + x1^2)]
[ -(r1^2 + x1^2)^(1/2) * v^2/(( -(r1^2 + x1^2)^(1/2) + r1)^2 + x1^2)]
```

舍去电阻值小于零的解，注意到$|Z_1| = \sqrt{R_1^2 + X_1^2}$，则$R = |Z_1|$时获得最大功率，其值为

$$P_{max} = \frac{|Z_1| V^2}{(|Z_1| + R_1)^2 + X_1^2}$$

例 8-13　正弦分析中元件值的求解 1

图 8-18 所示电路中，$R_1 = 10\ \Omega$，$C = 50\ \mu F$，$\omega = 200$ rad/s，电压表、电流表、功率表的读数分别为 30 V、1 A、15 W，求电阻 R 和电感 L 的值。

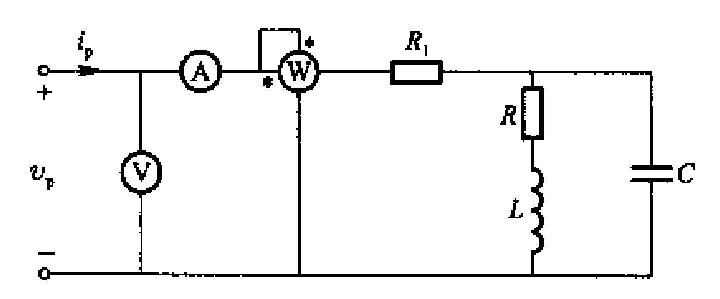


图 8-18　例 8-13 电路

解　由已知条件，$V_p = 30$ V，$I_p = 1$ A，$P = 15$ W，若令$\dot{I}_p = 1$ A，用符号表示电阻和电感的值，求出$\dot{V}_p$ 的表达式，应有

$$|\dot{V}_p| = V_p$$

$$\mathrm{Re}(\dot{V}_p) \times I_p = P$$

从以上两个等式关系可求出 R 和 L 的值。根据这一思想编写的 M 文件如下：

```
vp = 30; ip = 1; p = 15;
ckt = {'i 0 1 1'
```

```
        'rl 1 2 10'
        'r 2 3 R'
        'rl 3 0 j*200*L'
        'rc 2 0 1/(j*200*50e-6)'};
sana(ckt);
syms R L real
equ1 = abs(v1) - vp;
equ2 = real(v1) * ip - p;
[L,R] = solve(equ1,equ2);
R = numeric(R),L = numeric(L)
```

输出结果为

```
R =     3.1454
        9.0846
L =    0.10374
      -0.17243
```

由于电感必须取正值,故舍去第2组解,则

$$R = 3.1454\ \Omega$$

$$L = 0.10374\ \text{H}$$

例 8-14 正弦分析中元件值的求解 2

电路如图 8-19 所示,已知 $R_1 = 6\ \Omega$,$R_2 = 2\ \Omega$,$I = 1\ \text{A}$,$I_2 = 3\ \text{A}$,$\dot{I}$ 与 $\dot{V}$ 的相位差为 36.87°,求 ωL 和 $1/(\omega C)$。

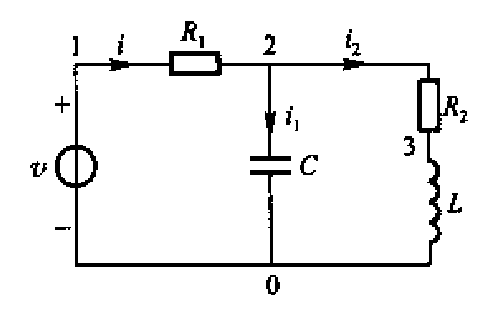


图 8-19 例 8-14 电路

解 若令 $X_L = \omega L$,$X_C = \dfrac{1}{\omega C}$(注:有些教材中定义 $X_C = -\dfrac{1}{\omega C}$),将电压源替换为电流源,$\dot{I} = 1\ \text{A}$,对电路列结点方程,由于电流 $\dot{I}_2$ 的模 $|\dot{I}_2|$ 为已知条件,列方

程时也可把 $\dot{I}_2$ 作为变量，一种比较简单的方法是：用电流控制电压源表示电阻 R_2，则该受控电压源的电流就是 $\dot{I}_2$。从结点方程解出 $\dot{I}_2$ 和 $\dot{V}$ 的表达式，根据已知条件

$$|\dot{I}_2| = 3\ \mathrm{A}$$

$$\mathrm{Im}(\dot{V})/\mathrm{Re}(\dot{V}) = \tan(-36.87°)$$

从以上两个方程即可解出 X_L 和 X_C。M 文件如下：

```
ckt = {'i 0 1 1'
       'r1 1 2 6'
       'rc 2 0  -j * xc'
       'e 2 3 e 2'
       '% resistor r2 is described with CCVS'
       'rl 3 0 j * xl'};
sana(ckt);
syms xc xl real
equ1 = abs(ie) - 3;
equ2 = imag(v1)/real(v1) - tan( -36.87 * pi/180);
[xc,xl] = solve(equ1,equ2);
xc = numeric(xc),xl = numeric(xl)
% validate the results
ie = subs(ie,{xl,xc},{'xl','xc'})
v1 = subs(v1,{xl,xc},{'xl','xc'})
i2mag = abs(ie),
v = numeric(v1);
vphase = degree(v)
```

该程序还对 X_L 和 X_C 的计算结果进行了验证。输出结果为

```
xc  =     7.027
        -11.527
xl  =     8.2462
         -8.2462
ie  =  -1.5616 -     2.5615i
        2.5616 +     1.5615i
v1  =       24 -         18i
```

```
          24 -          18i
i2mag =     3
            3
vphase =   -36.87
           -36.87
```

可看出,只有第1组解为正值,故

$$\frac{1}{\omega C}=7.027\ \Omega$$

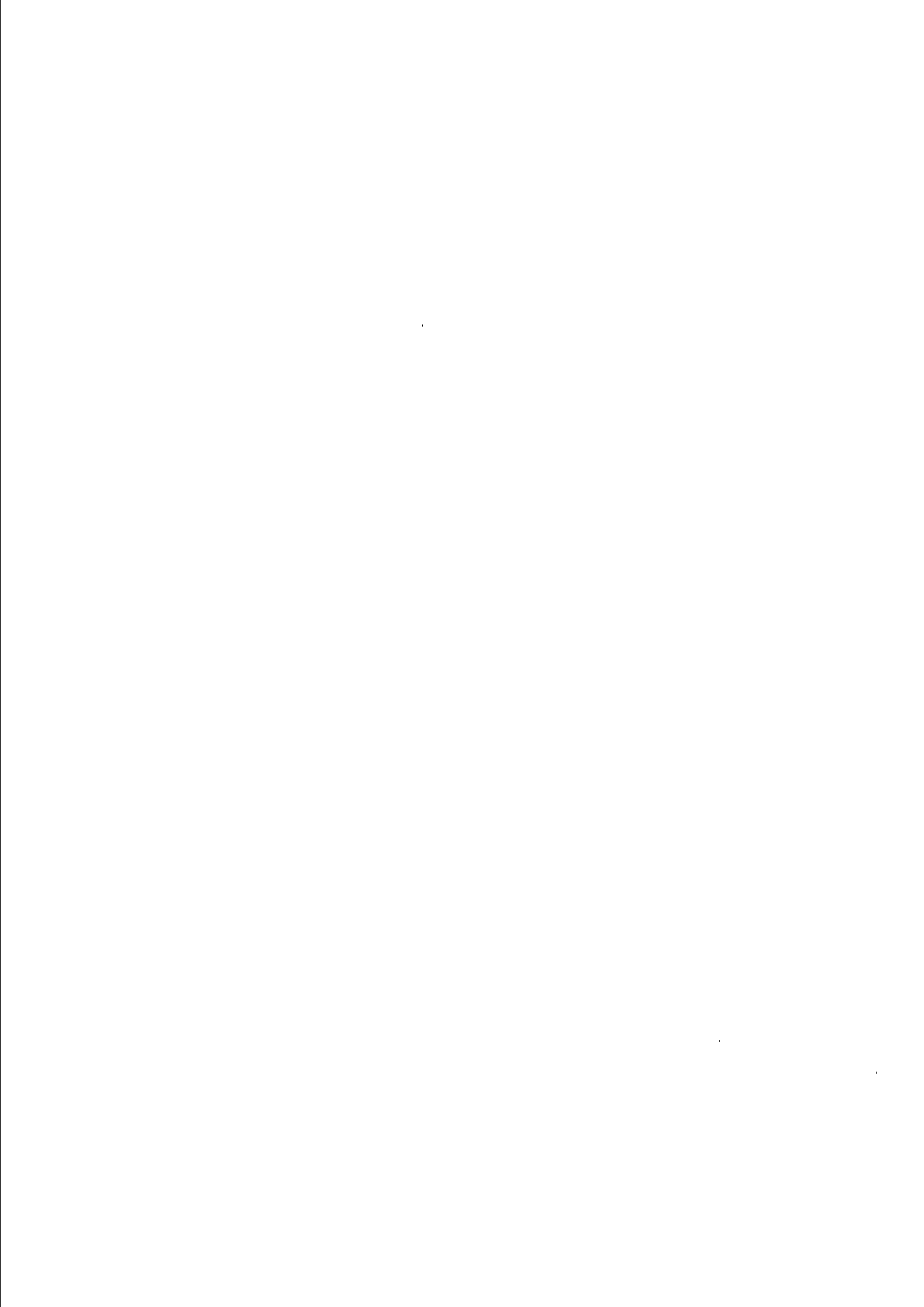
$$\omega L=8.246\ \Omega$$

以上两个例题说明,用电路符号分析函数 sana 和代数方程求解指令 solve 可确定元件的值。



第三篇

信号与系统分析



第九章 信号的可视化

信号通常用数学函数表示,一维信号按照自变量的取值是否连续分为连续时间信号和离散时间信号,分别用$x(t)$和$x[n]$表示。为了观察信号随时间变化的总体状况,信号常常用图形表示。MATLAB 强大的图形处理功能及符号运算功能为实现信号的可视化提供了强有力的工具。本章首先介绍用 MATLAB 实现连续时间信号和离散时间信号可视化的方法,然后介绍信号的自变量变换。

9.1 连续时间信号的可视化

严格说来,只有用符号解析式才能表示连续时间信号,而数值方法只是连续时间信号各个样点的数据,只有当样点取得非常密时才可近似看成连续时间信号。MATLAB 在绘制连续时间信号的曲线时,用直线连接相邻的两个样点,当取样间隔足够小时,这些离散的样点就能较好的表示连续时间信号。在 MATLAB 中连续时间信号可用数值和符号两种方式表示。

用数值方式表示连续时间信号$x(t)$,需要两个行向量 t 和 x 来分别代表给定的时间和信号的取值。定义时间取值的语句如下:

```
t = t1:p:t2
```

其中,t1 为信号的起始时间;t2 为终止时间;p 为时间间隔。若用 x 表示向量 t 所定义时间点上的函数值,利用 t 和 x 两个行向量,就可以利用 MATLAB 的绘图指令来绘出该信号的时域波形。

信号与系统中大多数信号都可以直接在 MATLAB 的函数库调用,详见第二章中表 2-7 的基本函数。对一些 MATLAB 函数库中没有定义的函数,例如阶跃函数,读者需要自己创建函数文件。

例 9-1 连续时间信号的波形

绘出符号函数 $\mathrm{sgn}(t)=\begin{cases}-1 & (t<0)\\ 1 & (t>0)\end{cases}$、sinc 函数 $\mathrm{sinc}(\pi t)=\frac{\sin(\pi t)}{\pi t}$①、指数函数 $e^{-1.5t}$ 和对数函数 $\ln t$ 的波形。

解 这几个函数都是 MATLAB 的函数库存在的函数,相应的语句为

符号函数 x1 = sign(t)

sinc 函数 x2 = sinc(t)

指数函数 x3 = exp(-1.5t)

对数函数 x4 = log(t)

M 文件如下:

```
t = -10:0.11:10;                 % 定义信号 x1 和 x2 的时间范围向量
x1 = sign(t);                    % 计算符号函数 x1
subplot(2,2,1);stairs(t,x1) ;    % 绘制符号函数波形
axis([ -10,10, -1.1,1.1]);       % 确定坐标系
x2 = sinc(t);                    % 计算 sinc 函数 x2
subplot(2,2,2);plot(t,x2)        % 绘制 sinc 函数波形
t =0:0.02:5; x3 = exp( -1.5 * t); % 计算指数函数 x3
subplot(2,2,3);plot(t,x3);       % 绘制指数函数波形
t =0.01:0.02:5;x4 = log(t);      % 计算对数函数 x4
subplot(2,2,4);plot(t,x4)        % 绘制对数函数波形
```

波形如图 9-1 所示。

例 9-2 单位阶跃函数的波形

绘出单位阶跃函数 $\varepsilon(t)$ 的波形和函数 $x(t)=\varepsilon(t)+\varepsilon(t-1)-2\varepsilon(t-2)$ 的波形。

解 单位阶跃函数 $\varepsilon(t)$ 在 MATLAB 函数库中不存在,需要用语句来实现。下面给出几种最简单的方法。

方法 1:利用阶跃函数与符号函数的关系 $\varepsilon(t)=0.5+0.5\,\mathrm{sgn}(t)$ 间接实现。

```
t = -1:0.001:2;x = 1/2 +0.5 * sign(t);
stairs(t,x);axis([ -1,2, -0.1,1.2])
```

方法 2:利用关系运算符实现。

① MATLAB 中定义 $\mathrm{sinc}(t)=\frac{\sin(\pi t)}{\pi t}$,而不是 $\mathrm{sinc}(\pi t)=\frac{\sin(\pi t)}{\pi t}$。

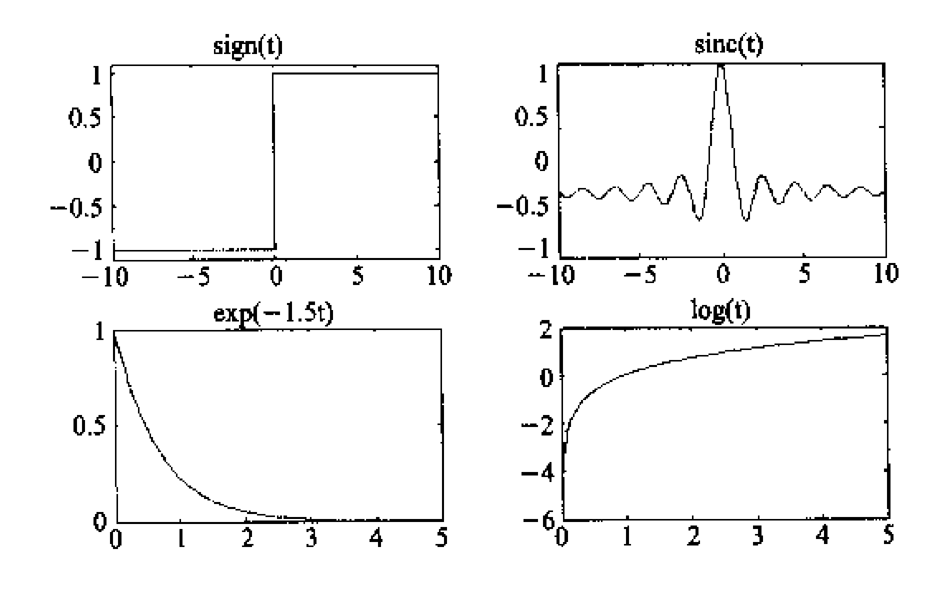


图 9－1 例 9－1 的波形

```
t = -1:0.001:2;x = (t>0); stairs(t,x);axis([-1,2,-0.1,1.2])
```

方法 3:建立一个函数文件(Heaviside.m),使用时可直接调用。

```
t = -1:0.001:2;x = Heaviside(t); stairs(t,x); axis([-1,2,-0.1,1.2])
function  x = Heaviside(t)
% unit step function
x = (t>0);
% end of function Heaviside
```

以上三种方法的结果相同,所绘出的波形均为图 9－2。

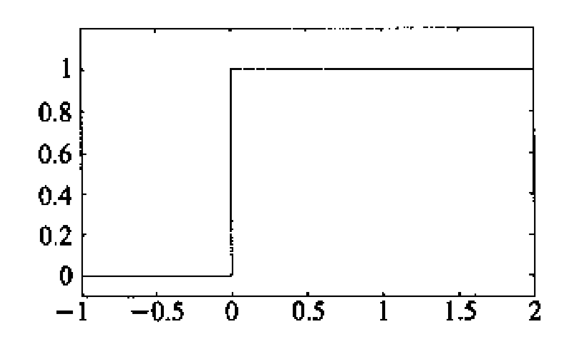


图 9－2 单位阶跃函数

函数 $x(t) = \varepsilon(t) + \varepsilon(t-1) - 2\varepsilon(t-2)$ 可以利用方法 2 来实现。

```
t = -1:0.005:3;x = (t>0) + (t>1) - 2*(t>2);
stairs(t,x);axis([-1,3,-0.1,2.2])
```

绘出的波形为图 9－3 所示。

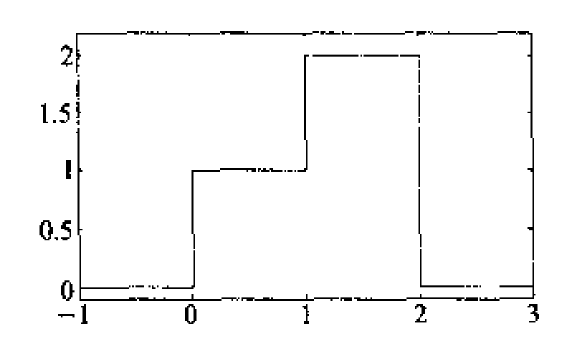


图 9-3 $x(t)=\varepsilon(t)+\varepsilon(t-1)-2\varepsilon(t-2)$

例 9-3 冲激函数的逼近

单位冲激函数 $\delta(t)$ 由于在 $t=0$ 处的值为无限大，用图形表示有一定困难，但一些函数在某一参数的极限时表现为冲激函数。下列三个函数在 $\tau\to0$ 时均为单位冲激函数，试绘出 τ 取 $1、2^4、3^4$ 时的波形。(1) 宽度为 τ，高度为 $\frac{1}{\tau}$ 的矩形脉冲；(2) 函数 $\frac{\sin(\pi t/\tau)}{\pi t}$；(3) 函数 $\frac{\tau}{\pi(\tau^2+t^2)}$。

解 (1) M 文件如下：

```
t = -1:0.001:1;
for i =1:3
    dt =1/(i^4);
    x =(1/dt) * ((t > = -(1/2 * dt)) -(t > =(1/2 * dt)));
    subplot(1,3,i);stairs(t,x);
end
```

波形如图 9-4 所示。

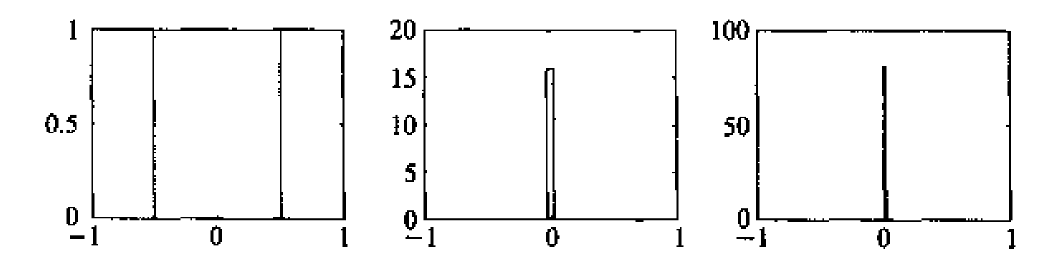


图 9-4 矩形脉冲逼近冲激函数的过程

(2) M 文件如下：

```
t = -10:0.011:10;
```

```
for i = 1:3
    dt = 1/(i^4);
    x = sin(pi * t/dt)./(pi * t);
    subplot(1,3,i);plot(t,x);
end
```

波形如图 9－5 所示。

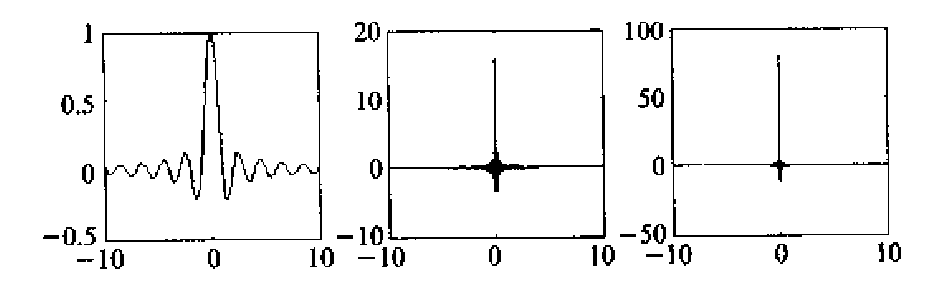


图 9－5 函数$\frac{\sin(\pi t/\tau)}{\pi t}$逼近冲激函数的过程

(3) M 文件如下：

```
t = -10:0.01:10;
for i = 1:3
    dt = 1/(i^4);
    x = dt./(pi.*(dt^2.+t.^2));
    subplot(1,3,i);plot(t,x);
end
```

波形如图 9－6 所示。

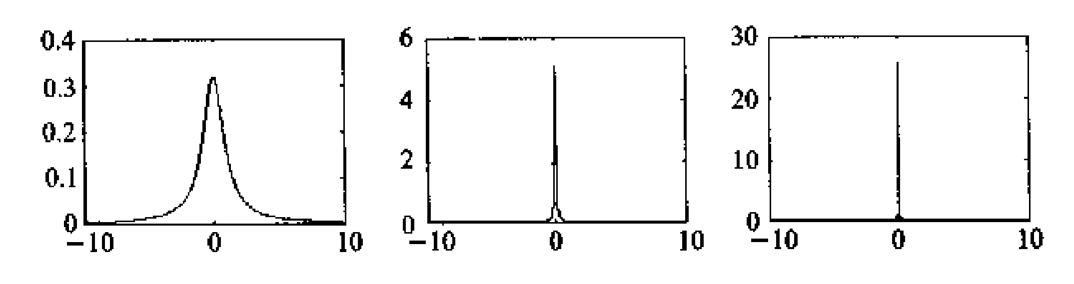


图 9－6 函数$\frac{\tau}{\pi(\tau^2+t^2)}$逼近冲激函数的过程

例 9－4 正弦信号的波形

在 $-10<t<10$ 的时间范围内绘出 $x_1(t)=\sin\left(t+\frac{\pi}{3}\right)$ 和 $x_2(t)=0.5\cos(2t)$ 的波形。

解　M 文件如下：

```
t = -10:0.01:10;x1 = sin(t+pi/3);
x2 =0.5 * cos(2 * t);
subplot(1,2,1);plot(t,x1);axis([-10,10, -1.1,1.2])
subplot(1,2,2);plot(t,x2);axis([-10,10, -0.6,0.6])
```

波形如图 9-7 所示。

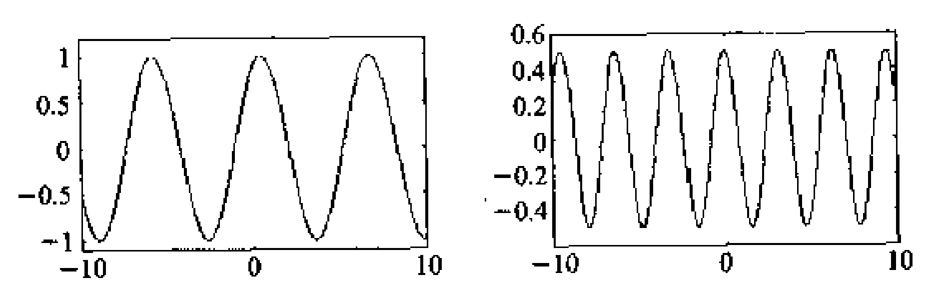


图 9-7　正弦余弦函数波形

周期矩形波函数的调用格式如下：

```
square(t)
square(t,duty)
```

该函数与 sin(t)类似,周期为 2π,振幅为 1(最大、最小值分别为 1 和 -1), t 为时间行向量。duty 为占空比的百分数,即函数值为正的区域在一个周期内所占的百分数,如果缺省,其值为 50,即在一个周期内函数值为正的持续时间等于函数值为负的持续时间。在$(0,2\pi\times duty/100)$区间函数值等于 1,在$(2\pi\times duty/100,2\pi)$区间函数值等于 -1 。

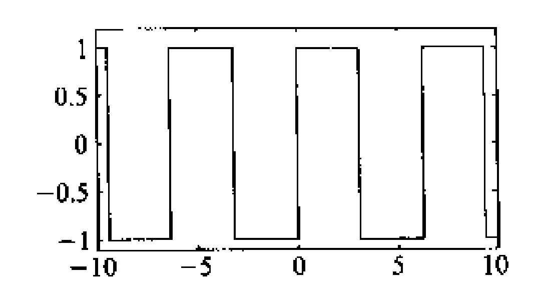
例如,以下语句的输出波形如图 9-8 所示。

```
t = -10:0.01:10;x1 =square(t); x2 =0.5 * (square(t,20) +1);
subplot(1,2,1);stairs(t,x1);axis([-10,10, -1.1,1.2]);
subplot(1,2,2);stairs(t,x2);axis([-10,10, -0.1,1.2])
```

周期锯齿波函数的调用格式如下：

```
sawtooth(t)
sawtooth(t,width)
```

其中,锯齿波的周期为 2π,振幅为 1, t 为时间行向量,width 指定最大值出现的



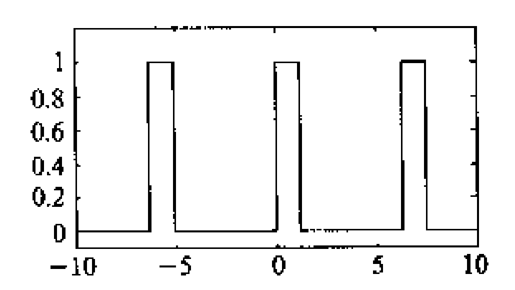
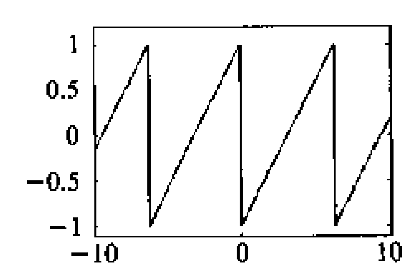


图 9-8 矩形波信号

位置,在0到1之间取值。当t由0增大到 width * 2π 时,函数值由 -1 增大到1,当t由 width * 2π 增大到 2π 时,函数值由 1 减小到 -1。例如:

```
t = -10:0.01:10;x1 = sawtooth(t); x2 =0.5 * (sawtooth(t,0.5) +1);
subplot(1,2,1);plot(t,x1);axis([-10,10,-1.1,1.2])
subplot(1,2,2);plot(t,x2);axis([-10,10,-0.1,1.2])
```

其波形如图 9-9 所示。



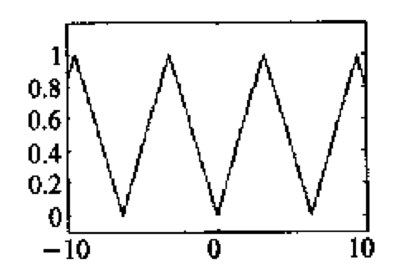


图 9-9 锯齿波和三角波

如果信号可以用一个符号表达式来表示它,就可以用 MATLAB 提供的 ezplot 指令绘制出信号的波形,注意符号法需要定义有关符号变量。

一般来讲,要用符号运算方法表示、计算和绘制信号波形,可以直接调用在 MATLAB 的符号工具箱中存在的符号函数。但应注意的是,函数 ezplot 只能画出既存在于符号工具箱中又存在于总 MATLAB 工具箱中的函数。因此,若要用 ezplot 画出仅存在于符号工具箱中函数波形,就需要用户在自己的工作目录 work 下创建一个同名同功能的函数文件。

例 9-5 信号的符号表达式及其波形

用符号表达式计算并绘出以下函数的波形。(1) $x_1(t) = \varepsilon(t)$;(2) $x_2(t) = \varepsilon(t) + \varepsilon(t-1) - 2\varepsilon(t-2)$;(3) $x_3(t) = \sin t$;(4) $x_4(t) =$

$\frac{t}{2}[\varepsilon(t)-\varepsilon(t-4)]$。

解 阶跃函数在符号工具箱中是 Heaviside 函数，它在总 MATLAB 工具箱中并不存在，所以必须先自己创建一个 Heaviside.m 函数文件（见例 9－2 的方法 3）。在工作目录 work 下创建 Heaviside.m 函数文件之后，就可以用 ezplot 绘波形了。

对于例 9－2 中用数值运算方法绘制的 $x_2(t)=\varepsilon(t)+\varepsilon(t-1)-2\varepsilon(t-2)$ 的波形，也可以用 Heaviside 函数的符号表达式来实现。

符号表达式的正弦函数与数值运算下的正弦函数一样都是 sin 函数，故可以直接调用。

函数 $x_4(t)=\frac{t}{2}[\varepsilon(t)-\varepsilon(t-4)]$ 可以很方便地用符号表达式来实现。

M 文件如下：

```
x1 = sym('Heaviside(t)');subplot(2,2,1);ezplot(x1,[-1,1])
x2 = sym('Heaviside(t) + Heaviside(t-1) -2 * Heaviside(t-2)');
subplot(2,2,2);ezplot(x2,[-1,3])
x3 = sym('3 * sin(0.5 * pi * t)');subplot(2,2,3);ezplot(x3)
x4 = sym('(t/2) * (Heaviside(t) - Heaviside(t-4))');
subplot(2,2,4);ezplot(x4,[-1,5])
```

波形如图 9－10 所示。

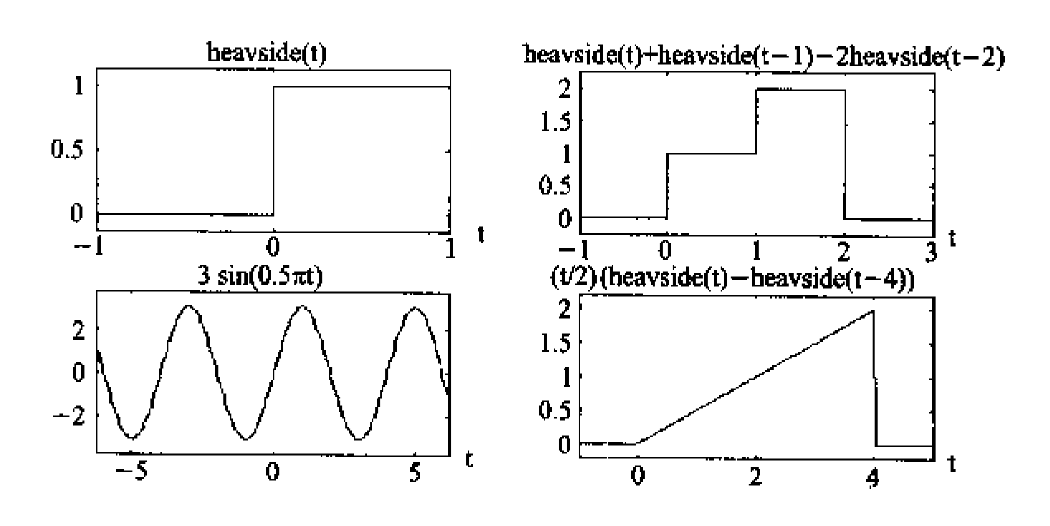


图 9－10 用符号运算方法绘制的函数波形

9.2 离散时间信号的可视化

离散时间信号也称为序列,用 $x[n]$ 表示,其中自变量 n 为整数,代表离散时间的序号。在用 MATLAB 表示序列并将其可视化时,应注意以下几点:(1) 与连续时间信号不同,离散时间信号无法用符号运算来表示;(2) 由于在 MATLAB 中,矩阵的元素个数是有限的,因此,MATLAB 无法表示无限长序列;(3) 在绘制离散时间信号的图形时,要使用枝干图绘制指令 stem,而不是 plot 指令;(4) 离散时间信号的自变量 n 为整数,即 n 的间隔为 1,可定义 $n=n1:\ n2$,起点 $n1$ 和终点 $n2$ 均为整数。(5) 连续时间周期信号离散化后,可能是周期的,也可能不再是周期的。

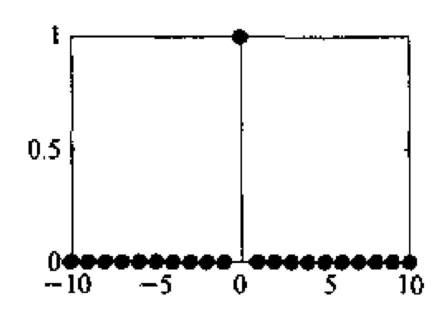
例 9-6 样值序列和阶跃序列的图形

绘出单位样值序列和单位阶跃序列的图形。

解 序列的起点和终点以交互方式输入,M 文件如下:

```
n1 = input('输入序列的起点 n1 = ');          % 以交互方式输入序列的起点 n1
n2 = input('输入序列的终点 n2 = ');          % 以交互方式输入序列的终点 n2
n = n1:n2;k = length(n);                    % 确定 n 向量及其元素的个数
x1 = zeros(1,k);x1(1, -n1 +1) =1;           % 实现单位样值序列
subplot(1,2,1);stem(n,x1,'filled')          % 绘制单位样值序列的图形
x2 = zeros(1,k);x2(1, -n1 +1:n2 -n1 +1) =1; % 实现单位阶跃序列
subplot(1,2,2);stem(n,x2,'filled')          % 绘制单位阶跃序列的图形
```

若运行该程序时按照提示输入:序列的起点 n1 = -10,序列的终点 n2 = 10,则可得到图 9-11 所示的图形。



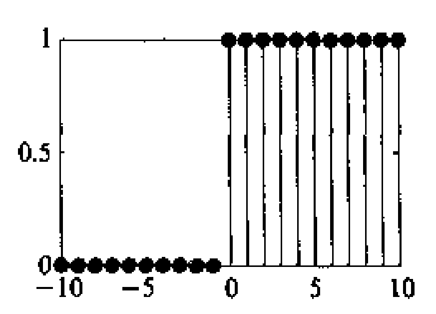


图 9-11 单位样值序列和单位阶跃序列

连续时间和离散时间正弦信号的一个重要差别是周期性问题。在连续时间

情况下，正弦信号 cos (ωt)是周期的，其周期等于 $2\pi/\omega$。在离散时间情况下，周期序列应满足

$$x[n]=x[n+N] \qquad \text{对全部 } n$$

其中，周期 N 必须是整数。用这个条件来验证离散时间正弦序列的周期性则有

$$\cos(\Omega_0 n+\varphi)=\cos(\Omega_0 n+\Omega_0 N+\varphi)$$

这要求

$$\Omega_0 N=2k\pi$$

或

$$\frac{\Omega_0}{2\pi}=\frac{k}{N}$$

式中 k 为整数。这就要求 $\frac{\Omega_0}{2\pi}$ 必须为有理分式，若 Ω_0 的取值不满足此关系，正弦序列就不是周期的。

例 9-7　序列的周期性

分析正弦序列 cos ($\Omega_0 n$)的周期性、振荡快慢与频率 Ω_0 的关系。

解　令 Ω_0 取不同的值，观察正弦序列的周期性。结果如图 9-12 所示。

```
n = -20:20;
x1 = cos(2 * pi * n);subplot(3,3,1);stem(n,x1,'filled'),
title('2\pi or 0'),line([-20,20],[0 0])
x2 = cos(pi/8 * n);subplot(3,3,2);stem(n,x2,'filled'),
title('\pi/8'),line([-20,20],[0 0])
x3 = cos(pi/4 * n);subplot(3,3,3);stem(n,x3,'filled'),
title('\pi/4'),line([-20,20],[0 0])
x4 = cos(pi/2 * n);subplot(3,3,4);stem(n,x4,'filled'),
title('\pi/2'),line([-20,20],[0 0])
x5 = cos(pi * 4/5 * n);subplot(3,3,5);stem(n,x5,'filled'),
title('4\pi/5'),line([-20,20],[0 0])
x6 = cos(pi * n);subplot(3,3,6);stem(n,x6,'filled'),
title('\pi'),line([-20,20],[0 0])
x7 = cos(15 * pi/8 * n);subplot(3,3,7);stem(n,x7,'filled'),
title('15\pi/8'),line([-20,20],[0 0])
x8 = cos(n);subplot(3,3,8);stem(n,x8,'filled'),
title('1'),line([-20,20],[0 0])
```

```
x9 = cos(5/11 * n);subplot(3,3,9);stem(n,x9,'filled'),
title('5/11'),line([-20,20],[0 0])
```

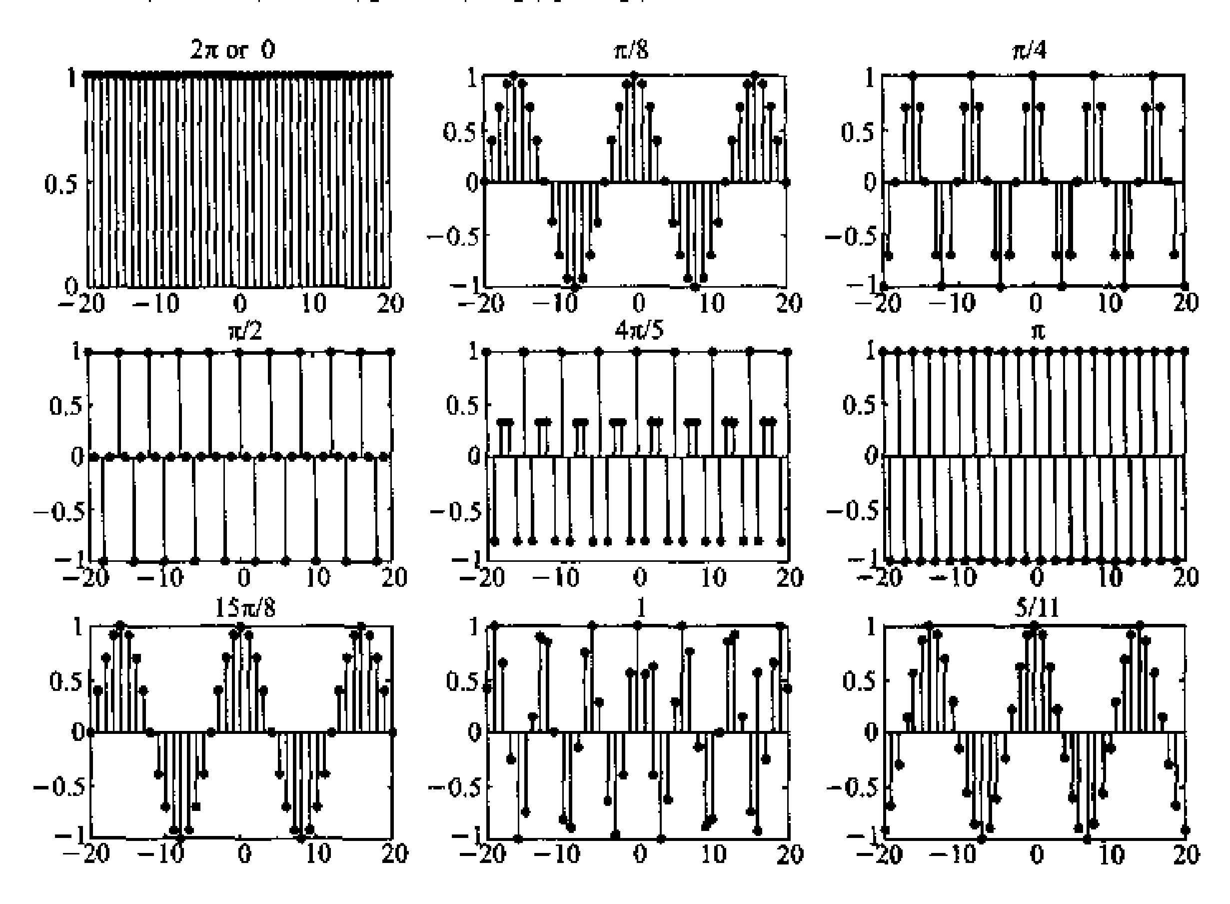


图 9-12 Ω_0 取不同值时的正弦序列

从图 9-12 可以看出：当 Ω_0 取 1 和 5/11 等数值时，序列不是周期的，这是因为 $\Omega_0/2\pi$ 不是有理数；当 Ω_0 从 0 增大到 π 时，正弦周期序列的振荡加快，而 Ω_0 从 π 增大到 2π 时，正弦周期序列的振荡放慢。

9.3 连续时间信号的自变量变换及运算

对于连续时间信号，MATLAB 有数值和符号两种表示方法。这两种方法均可实现信号的自变量变换和信号的运算，但使用符号方法则比较方便，因为在 MATLAB 中有专门的函数用于自变量变换和信号运算。

信号的自变量变换包括信号的移位、反转和尺度变换。在 MATLAB 的符号运算中，它们都可以用 subs 指令来实现操作。

信号的移位也称平移，对连续时间信号 $x(t)$，若有常数 $t_0>0$，信号 $x(t-t_0)$

是把 $x(t)$ 沿 t 轴方向平移时间 t_0；而信号 $x(t+t_0)$ 是把 $x(t)$ 沿 t 轴的相反方向平移时间 t_0。指令为

```
x1 = subs(x,t,t-t0)
x2 = subs(x,t,t+t0)
```

信号的反转是指将信号以纵坐标轴反转，即将信号 $x(t)$ 中的自变量 t 换为 $-t$。指令为

```
x3 = subs(x,t,-t)
```

信号的尺度变换是指将信号的横坐标进行展宽和压缩变换，即将信号 $x(t)$ 中的自变量 t 换为 at。当 $a>1$ 时，信号 $x(at)$ 以原点为基准，沿横轴压缩到原信号 $x(t)$ 的 $1/a$；当 $0<a<1$ 时，信号 $x(at)$ 以原点为基准，沿横轴展宽至原信号 $x(t)$ 的 $1/a$ 倍。指令为

```
x4 = subs(x,t,a*t)
```

例 9-8　信号的自变量变换

已知连续时间信号 $x(t)=(4-t)[\varepsilon(t)-\varepsilon(t-4)]$，绘出 $x_1(t)=x(t-2)$、$x_2(t)=x(t+2)$、$x_3(t)=x(-t)$、$x_4(t)=x(2t)$、$x_5(t)=x(t/2)$ 信号的波形。

解　用符号运算方法实现各种自变量变换，M 文件如下：

```
syms t;                                      % 定义符号变量 t
x = sym('(4-t)*(Heaviside(t)-Heaviside(t-4))')
% 信号 x 的符号表达式
subplot(2,3,1);ezplot(x,[-4,9]);             % 绘制 x 的波形
x1 = subs(x,t,t-2)                           % 信号 x1
subplot(2,3,2);ezplot(x1,[-4,9]);            % 绘制 x1 的波形
x2 = subs(x,t,t+2)                           % 信号 x2
subplot(2,3,3);ezplot(x2,[-4,9]);            % 绘制 x2 的波形
x3 = subs(x,t,-t)                            % 信号 x 的反转
subplot(2,3,4);ezplot(x3,[-4,9]);            % 绘制 x3 的波形
x4 = subs(x,t,2*t)                           % 信号 x4
subplot(2,3,5);ezplot(x4,[-4,9]);            % 绘制 x4 的波形
x5 = subs(x,t,t/2)                           % 信号 x5
subplot(2,3,6);ezplot(x5,[-4,9]);            % 绘制 x5 的波形
```

各信号的波形如图 9-13 所示。

信号的时域运算包括信号的相加、相乘等运算。在 MATLAB 的符号运算中，只要事先定义参加运算的信号 $x_1(t)$ 和 $x_2(t)$ 为符号变量，它们就可以直接

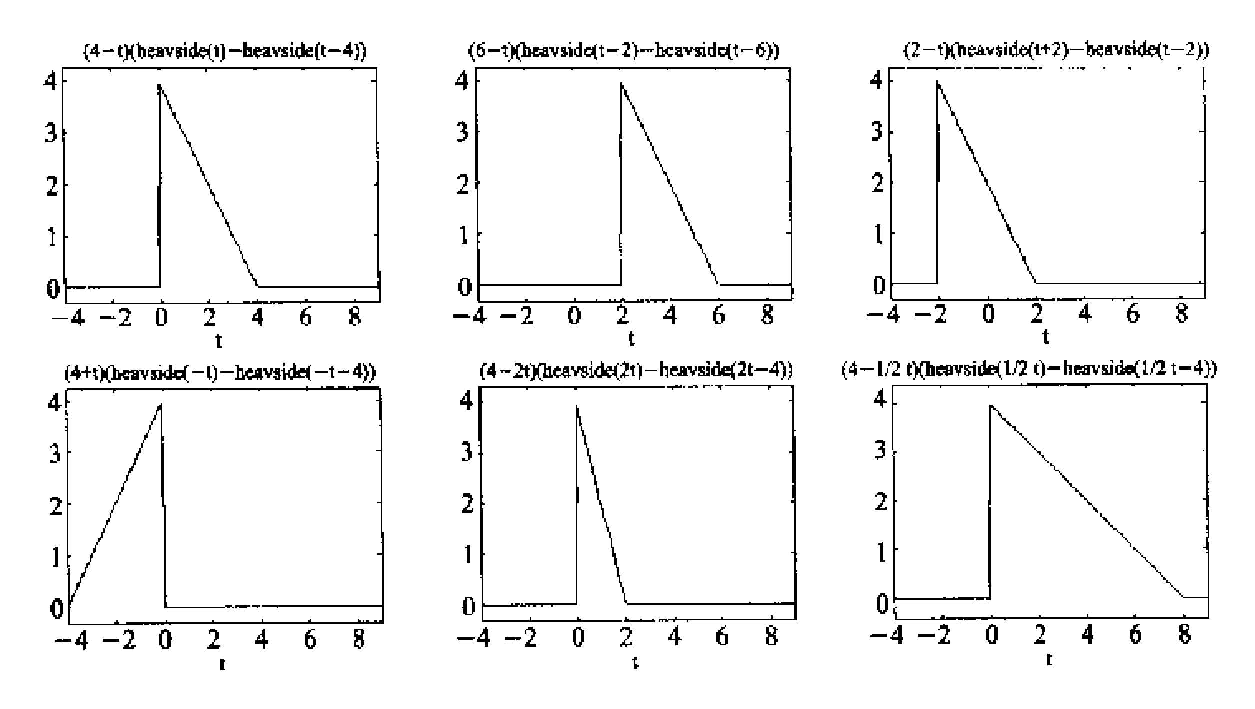


图 9－13 信号 $x(t)=(4-t)[\varepsilon(t)-\varepsilon(t-4)]$ 的各种自变量变换

利用基本运算符进行计算。具体指令为

```
x = x1 + x2   % 把两个符号函数 x1 和 x2 的对应时刻值相加
y = x1 * x2   % 把两个符号函数 x1 和 x2 的对应时刻值相乘
```

例 9－9 信号的相加和相乘

已知信号 $x_1(t)=e^{-2t}\varepsilon(t)$ 及信号 $x_2(t)=\sin(2\pi t)$，绘出 $x_3(t)=x_1(-t)$、$x_4(t)=x_1(t)+x_3(t)$、$x_5(t)=x_1(t)x_2(t)$、$x_6(t)=x_4(t)x_2(t)$ 的波形。

解 用符号运算方法实现。波形如图 9－14 所示。

```
syms t                                      % 定义符号变量 t
x1 = sym('exp( -2 * t) * Heaviside(t) ');   % 计算符号函数 x1
subplot(2,3,1);ezplot(x1,[ -3,3]);          % 绘制 x1 的波形
x2 = sin(2 * pi * t);
subplot(2,3,2);ezplot(x2,[ -3,3]);          % 绘制 x2 的波形
x3 = subs(x1,t, -t);
subplot(2,3,3);ezplot(x3,[ -3,3]);
x4 = x1 + x3;
subplot(2,3,4);ezplot(x4,[ -3,3]);
x5 = x1 * x2;
subplot(2,3,5);ezplot(x5,[ -3,3]);
```

```
x6 = x4 * x2;
subplot(2,3,6);ezplot(x6,[ -3,3]);
```

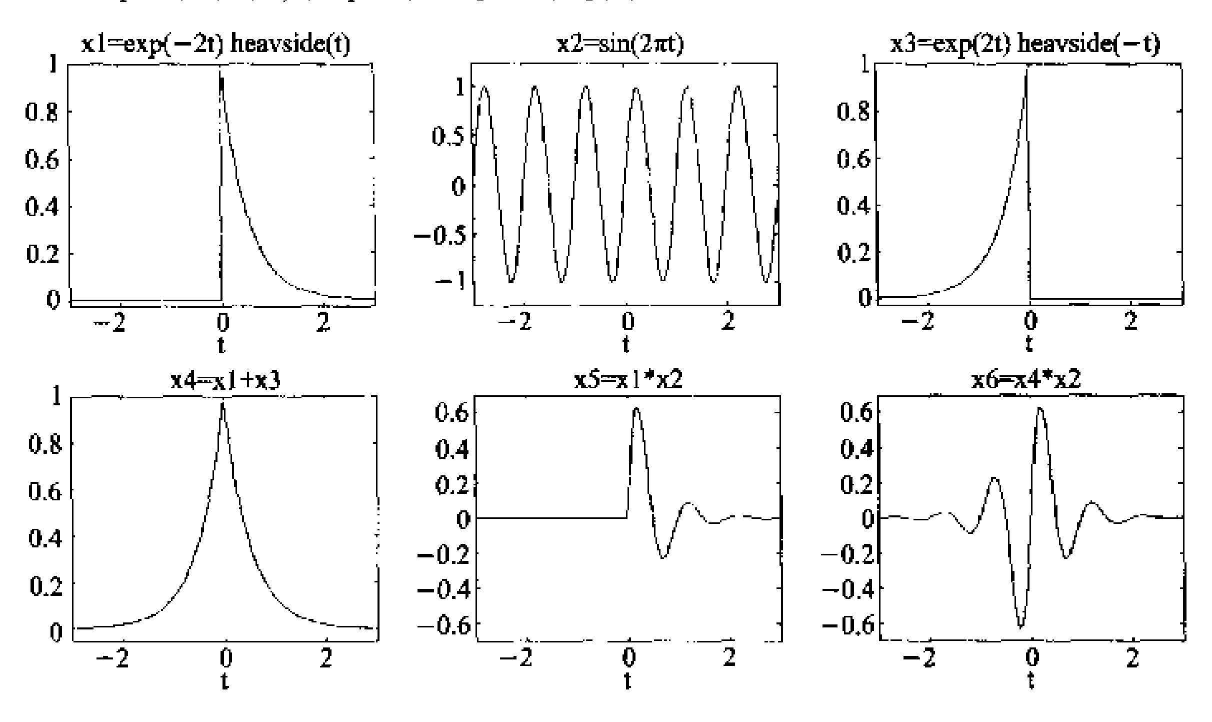


图 9－14　信号的时域相加、相乘运算

连续时间信号 $x(t)$ 可以分解为偶对称和奇对称分量,其计算公式为:

$$x_e(t)=\frac{1}{2}[x(t)+x(-t)]$$

$$x_o(t)=\frac{1}{2}[x(t)-x(-t)]$$

利用上述对信号的时域操作方法,可以很方便地实现信号的奇偶分解。

例 9－10　信号的偶对称和奇对称分量

分解例 9－9 中信号 $x_1(t)=e^{-2t}\varepsilon(t)$ 的奇偶分量。

解　根据计算公式,M 文件如下:

```
syms t
x1 = sym('exp( -2 * t) * Heaviside(t)');
x2 = subs(x1,t, - t);                     % 信号 x1 的反转
x3 =0.5 * (x1 +x2);                       % 信号 x1 的偶分量
subplot(1,2,1);ezplot(x3,[ -3,3]);        % 绘制信号 x1 的偶分量波形
x4 =0.5 * (x1 - x2);                      % 信号 x1 的奇分量
subplot(1,2,2);ezplot(x4,[ -3,3]);        % 绘制信号 x1 的奇分量波形
```

波形如图 9-15 所示。

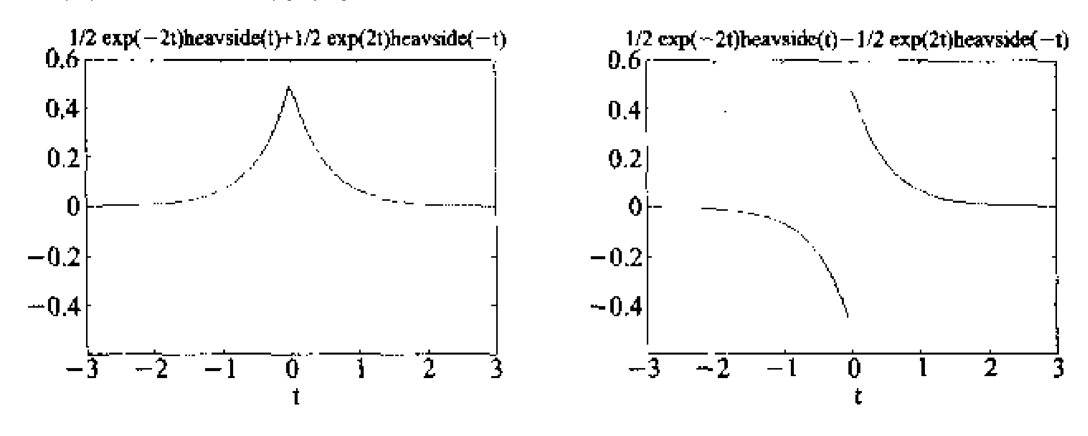


图 9-15 信号的奇偶分解

9.4 离散时间信号的自变量变换及运算

离散时间信号的自变量变换及运算与连续时间信号不同，它不能用符号运算来实现，必须用数值运算的方法。因为 MATLAB 的数值运算方法中没有专门的自变量变换函数，所以，需要用户自己编制函数文件来实现序列各个不同的时域变换及运算。在使用这些函数文件时，应注意把它们放在当前工作目录下。

序列 $x[n]$ 的移位可看作是将序号向量 n 移位（平移），而表示对应时间序号点的序列样值不变，当序列向左移动 n_0 个单位时，n 向量的所有时间序号都减小 n_0 个单位，反之则增加 n_0 个单位。可用自编的函数文件 yiwei.m 来实现序列的移位。

```
function [x1,n1] = yiwei(x,n,n0)
% n1 = n + n0
n1 = n + n0;
x1 = x;
```

序列 $x[n]$ 的反转即是将表示序列的向量 x 和 n 以零时刻取值为基准点，以纵轴为对称轴反转。可用自编的函数文件 fanzhuan.m 来实现序列的反转。

```
function [x1,n1] = fanzhuan(x,n)
n1 = -fliplr(n);
x1 = fliplr(x);
```

序列 $x[n]$ 的尺度变换与连续时间信号 $x(t)$ 的尺度变换不完全相同。因为自变量的向量 n 中元素的取值只能是整数，所以当 a 为大于 1 的正整数时，压缩

后的序列 $x[an]$ 要丢失原序列 $x[n]$ 的一些信息，而扩展后的序列 $x[n/a]$ 要填入必要的零值。可用自编的函数文件 yasuo.m 来实现序列的压缩。编写的压缩函数文件如下：

```
function [x1,n1] = yasuo(x,n,a)
% 实现序列 x[n]的压缩
n2 = length(n);
x1(1,1) = x(1,1);
k = 1;
while k < (n2/a)
    k = k + 1;
    x1(1,k) = x(1,a * (k - 1) + 1);
end
n1 = 0:(length(x1) - 1);
```

用自编的函数文件 kuozhan.m 来实现序列的扩展，函数文件如下：

```
function [x1,n1] = kuozhan(x,n,a)
% 实现离散时间信号 x[n]的扩展 x[n/a]
n2 = length(n);
x1(1,1) = x(1,1);
k = 1;k1 = 1;
x1 = zeros(1,n2 * a);
while k < n2
    k1 = k1 + a;k = k + 1;
    x1(1,k1) = x(1,k);
end
n1 = 0:(length(x1) - 1);
```

例 9-11　序列的自变量变换

已知 $x[n]=[\underset{\substack{\uparrow \\ n=0}}{0},1,2,-1,-2,1,3,4,4]$，绘制 $x[n-2]$、$x[n+2]$、$x[-n]$、$x[n/2]$、$x[2n]$ 的图形。

解　利用上面自编的几个函数文件，可以很方便地实现序列的各种自变量变换。M 文件如下：

```
n = 0:8;
x = [0,1,2,-1,-2,1,3,4,4];subplot(2,3,1);      % 序列 x[n]
```

```
stem(n,x,'filled');axis([-8,18,-3,5]);title('x[n]');
[x1,n1] = yiwei(x,n,2);subplot(2,3,2);            % x[n-2]
stem(n1,x1,'filled');axis([-8,18,-3,5]);title('x[n-2]')
[x2,n2] = yiwei(x,n,-2);subplot(2,3,3);           % x[n+2]
stem(n2,x2,'filled');axis([-8,18,-3,5]);title('x[n+2]')
[x3,n3] = fanzhuan(x,n);subplot(2,3,4);           % x[-n]
stem(n3,x3,'filled');axis([-8,18,-3,5]);title('x[-n]')
[x4,n4] = kuozhan(x,n,2);subplot(2,3,5);          % x[n/2]
stem(n4,x4,'filled');axis([-8,18,-3,5]);title('x[n/2]')
[x5,n5] = yasuo(x,n,2);subplot(2,3,6);            % x[2n]
stem(n5,x5,'filled');axis([-8,18,-3,5]);title('x[2n]')
```

波形如图 9-16 所示。

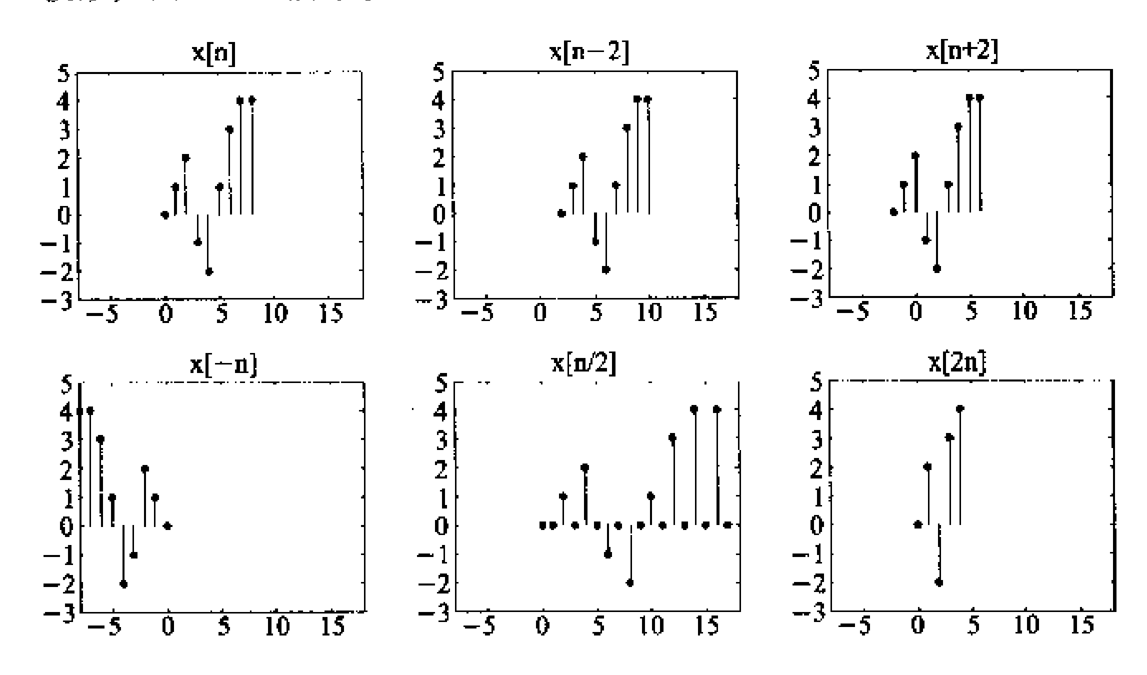


图 9-16 离散时间信号 x[n]的各种自变量变换

在 MATLAB 中,序列的相加和相乘运算需表示成向量的相加和相乘,因此要求参加运算的序列一定要具有相同的长度。对于不同长度的两个序列不能直接进行运算,必须先通过补零的方法使各序列具有相同的长度。要实现对任意长度的两个序列相加和相乘运算,也需要用户自己编制函数文件。

可用自编的函数文件 jia.m 来实现两序列的相加。函数文件如下:

```
function [y,n] = jia(x1,n1,x2,n2)
```

```
% y[n] = x1[n] + x2[n]
n = min(min(n1),min(n2)):max(max(n1),max(n2));
s1 = zeros(1,length(n));s2 = s1;
s1(find((n>=min(n1))&(n<=max(n1))==1)) = x1;
s2(find((n>=min(n2))&(n<=max(n2))==1)) = x2;
y = s1 + s2;
```

可用自编的函数文件 cheng.m 来实现两序列的相乘，函数文件如下：

```
function [y,n] = cheng(x1,n1,x2,n2)
% y[n] = x1[n] * x2[n]
n = min(min(n1),min(n2)):max(max(n1),max(n2));
s1 = zeros(1,length(n));s2 = s1;
s1(find((n>=min(n1))&(n<=max(n1))==1)) = x1;
s2(find((n>=min(n2))&(n<=max(n2))==1)) = x2;
y = s1.* s2;
```

例 9-12 序列的相加和相乘

已知 $x_1[n]=[\underset{n=0}{\underset{\uparrow}{1}},0,2,4,3,-1,2,3]$，$x_2[n]=[\underset{n=3}{\underset{\uparrow}{2}},3,1,2,4,3]$，用 MATLAB 绘出下列信号的图形。(1) $x_1[n]$；(2) $x_1[n]$的偶对称和奇对称分量；(3) $x_1[n]x_2[n]$。

解 利用上面给出的函数文件，编写的 M 文件如下：

```
n1 = 0:7;x1 = [1,0,2,4,3,-1,2,3];subplot(2,3,1);        % 定义 x1
stem(n1,x1,'filled');axis([-8,8,-4,5]);title('x1[n]');
n2 = 3:8;x2 = [2,3,1,2,4,3];subplot(2,3,2);              % 定义 x2
stem(n2,x2,'filled');axis([-8,8,-4,5]);title('x2[n]');
[y,n] = fanzhuan(x1,n1);subplot(2,3,3);                  % x1 的反转
stem(n,y,'filled');axis([-8,8,-4,5]);title('y[n] = x1[-n]');
[x3,n3] = jia(0.5*x1,n1,+0.5*y,n);subplot(2,3,4);
% 计算 x1 的偶分量
stem(n3,x3,'filled');axis([-8,8,-4,5]);
title('(1/2)(x[n]+x[-n])');
[x4,n4] = jia(0.5*x1,n1,-0.5*y,n);subplot(2,3,5);
% 计算 x1 的奇分量
stem(n4,x4,'filled');axis([-8,8,-4,5]);
```

```
title('(1/2)(x[n] - x[ - n])');
[x5,n5] = cheng(x1,n1,x2,n2);subplot(2,3,6);          % 计算 x1x2
stem(n5,x5,'filled');axis([ -8,8, -4,15]);title('x1[n] * x2[n]');
```

图形如图 9-17 所示。

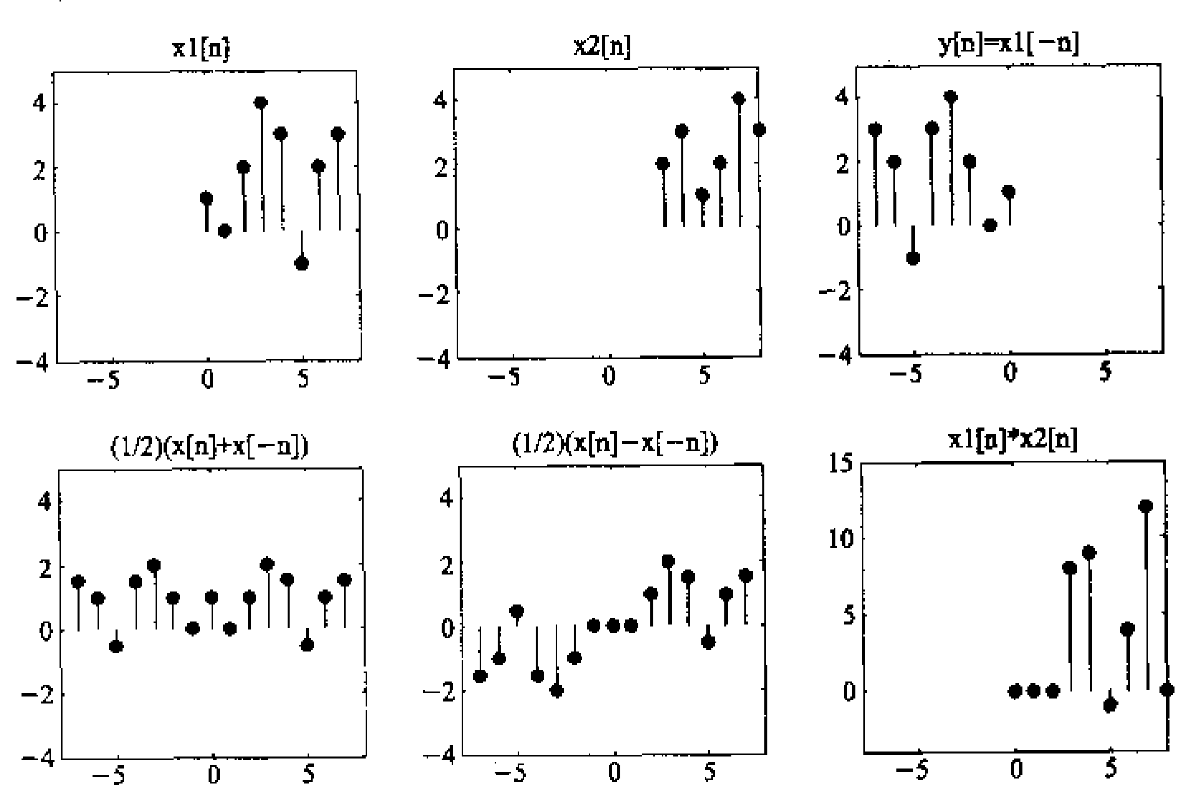


图 9-17 离散时间信号的相加与相乘

第十章　LTI 系统的时域数值分析

本章介绍线性时不变(LTI)系统时域数值分析的指令,内容包括:输入输出微分方程的求解指令,差分方程的递推求解,卷积求和指令与数值卷积方法,连续时间状态空间方程及其求解,离散时间状态空间方程及其求解。有关输入输出微分方程的符号求解和变换求解内容见第七章和第十一章,差分方程的变换解法见第十二章。

10.1　连续时间系统的时域分析

对连续时间 LTI 系统,时域输入输出方程的一般形式为

$$a_0\frac{\mathrm{d}^N y(t)}{\mathrm{d}t^N}+a_1\frac{\mathrm{d}^{N-1}y(t)}{\mathrm{d}t^{N-1}}+\cdots+a_N y(t)=$$

$$b_0\frac{\mathrm{d}^M x(t)}{\mathrm{d}t^M}+b_1\frac{\mathrm{d}^{M-1}x(t)}{\mathrm{d}t^{M-1}}+\cdots+b_M x(t) \tag{10-1}$$

其中,$x(t)$为输入信号,$y(t)$为输出信号。

线性系统的响应是由初始状态和输入共同作用的结果。当初始状态为零时,系统的响应称为零状态响应;当输入为零时,系统的响应称为零输入响应;当初始状态和输入均不为零时,系统的响应称为全响应,它可分解为零状态响应和零输入响应的和。

在零初始状态下,如果系统的输入信号为单位冲激函数,其响应称为(单位)冲激响应,如果系统的输入信号为单位阶跃函数,其响应称为(单位)阶跃响应。

MATLAB 求解式(10-1)的一些指令如下:

(1) 冲激响应。

```
impulse(b,a)
impulse(b,a,t)
[y,t] = impulse(b,a)
```

其中,a,b 表示系统的有关系数向量 $\boldsymbol{a},\boldsymbol{b}$,$\boldsymbol{a}=[a_0,a_1,\cdots,a_N]$,$\boldsymbol{b}=[b_0,b_1,\cdots,$

b_M];t 为离散时间行向量;y 为输出信号的样本。当输出变元不给出时,该函数在屏幕上绘制出 $t \geq 0_+$ 时的冲激响应曲线。当输出信号在 $t=0$ 处含有冲激时(此时 $M \geq N$),注意输出曲线不能反映其冲激分量。

(2) 阶跃响应。

```
step(b,a)
step(b,a,t)
[y,t] = step(b,a)
```

(3) 任意输入的零状态响应。

```
lsim(b,a,x,t)
[y,t] = lsim(b,a,x,t)
```

其中,x 为输入信号的样本。

一些实周期信号的样本 x 可用 gensig 函数生成,其格式为

```
[x,t] = gensig(type,T)
[x,t] = gensig(type,T,Tf,Ts)
```

其中,T 为信号的周期;Tf 为信号截取的长度或终止时间;Ts 为取样间隔;type 用于指定信号的类型:正弦波、方波、矩形脉冲,分别为

```
type = 'sin'
type = 'square'
type = 'pulse'
```

以上三种信号的振幅均为 1 。

MATLAB 没有提供求解式(10-1)零输入响应的指令,读者可用本书作者提供的 initialr 函数求解,其原理将在 10.5 节介绍。该函数的调用格式如下:

```
initialr(b,a,y0)
initialr(b,a,y0,t)
[y,t] = initialr(b,a,y0)
```

其中,y0 表示初始状态的行向量,$\boldsymbol{y}_0 = \left[\left.\frac{\mathrm{d}^{N-1}y(t)}{\mathrm{d}t^{N-1}}\right|_{t=0}, \left.\frac{\mathrm{d}^{N-2}y(t)}{\mathrm{d}t^{N-2}}\right|_{t=0}, \cdots, y(0) \right]$。

将零状态响应与零输入响应叠加,即可得到全响应。

例 10-1 冲激响应和阶跃响应

已知系统的输入输出微分方程为

$$\frac{\mathrm{d}^2y(t)}{\mathrm{d}t^2} + 5\frac{\mathrm{d}y(t)}{\mathrm{d}t} + 6y(t) = 2\frac{\mathrm{d}x(t)}{\mathrm{d}t} + x(t)$$

求单位冲激响应和单位阶跃响应。

解　M 文件如下：

```
a = [1,5,6];
b = [2,1];
impulse(b,a)
figure(2)
step(b,a)
```

输出波形如图 10－1 和图 10－2 所示。

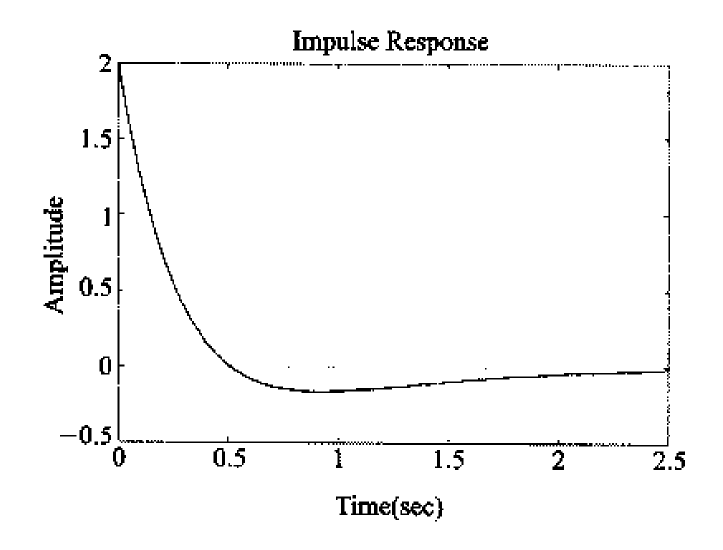


图 10－1　冲激响应

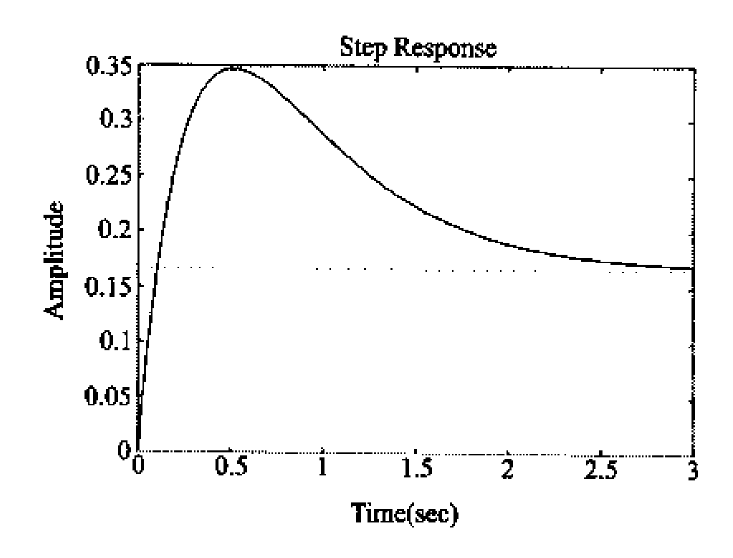


图 10－2　阶跃响应

例 10－2 零状态响应

上例中，若 $x(t)=\sin\left(\frac{2\pi}{T}t\right)\varepsilon(t)$，周期 $T=0.5$ s，求零状态响应。

解 M 文件如下：

```
a = [1,5,6];
b = [2,1];
[x,t] = gensig('sin',0.5,3);
lsim(b,a,x,t)
grid
```

输出波形如图 10－3 所示。

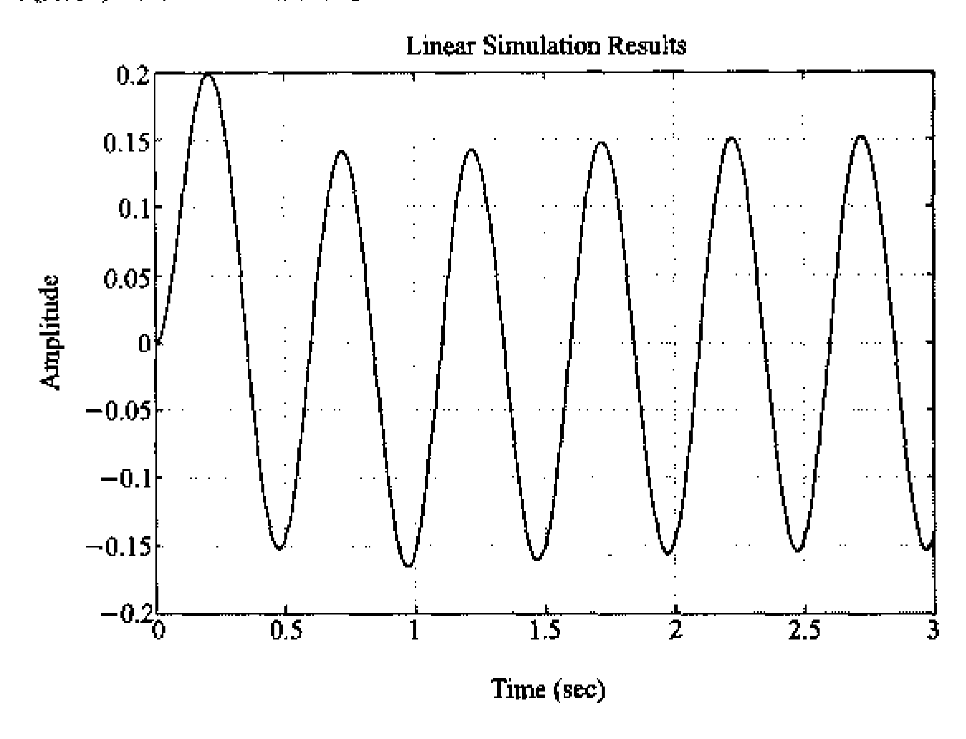


图 10－3 例 10－2 的输出波形

例 10－3 全响应

设 $\frac{d^2y(t)}{dt^2}+5\frac{dy(t)}{dt}+6y(t)=2\frac{dx(t)}{dt}+x(t)$，$x(t)=6\varepsilon(t)$，$y'(0_-)=10$，$y(0_-)=0$，求 $y(t)$ 的零状态响应、零输入响应和全响应。

解 零输入响应用 initialr 指令计算，零状态响应用 lsim 指令计算，全响应为零输入响应与零状态响应的和。M 文件如下：

```
a = [1,5,6];
b = [2,1];
y0 = [10,0];
```

```
[yzi,t] = initialr(b,a,y0);
x = 6 * ones(1,length(t));
yzs = lsim(b,a,x,t);
y = yzi + yzs;
plot(t,yzi,'-.',t,yzs,'--',t,y,'-')
legend('zero input','zero state','complete response')
grid
```

输出波形如图 10-4 所示。

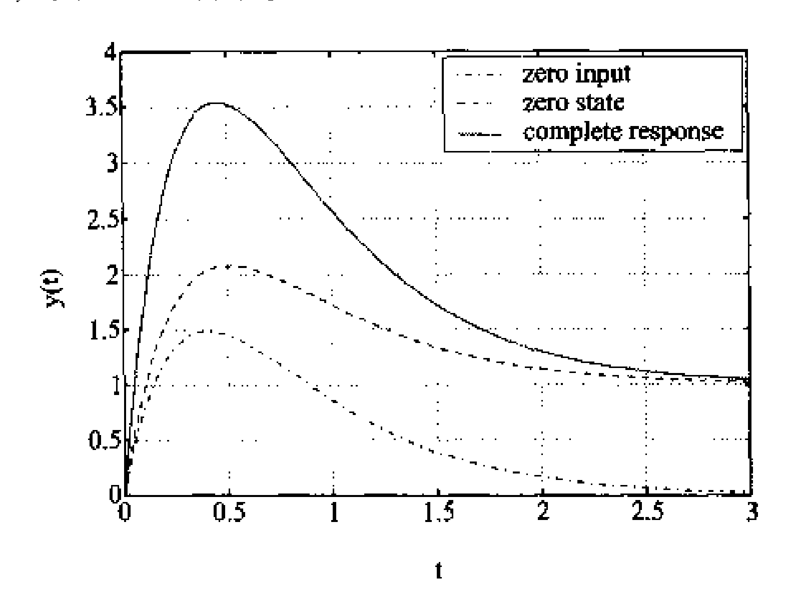


图 10-4 例 10-3 的输出波形

10.2 离散时间系统的时域分析

单输入单输出离散时间系统用差分方程描述，即

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \qquad (10-2)$$

其中，$x[n]$为输入；$y[n]$为输出。不失一般性，设 $a_0=1$，则 $y[n]$可表示为

$$y[n] = -\sum_{k=1}^{N} a_k y[n-k] + \sum_{k=0}^{M} b_k x[n-k] \qquad (10-3)$$

即

$$\begin{aligned} y[n] = & -a_1 y[n-1] - a_2 y[n-2] - \cdots - a_N y[n-N] + \\ & b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \end{aligned} \qquad (10-4)$$

显然，当 $y[-1]$、$y[-2]$、…、$y[-N]$ 以及 $x[n]$ 给定情况下，用递推方法可求出 $y[0]$，$y[1]$，…。式(10-4)也可写成

$$y[n]=-[a_1,a_2,\cdots,a_N]\begin{bmatrix}y[n-1]\\y[n-2]\\\vdots\\y[n-N]\end{bmatrix}+[b_0,b_1,\cdots,b_M]\begin{bmatrix}x[n]\\x[n-1]\\\vdots\\x[n-M]\end{bmatrix} \tag{10-5}$$

根据式(10-5)编制 MATLAB 程序时，由于数组的下标只能取正数，因此，要注意数组与序号的对应关系。如果输入 $x[n]$ 为因果序列，求解差分方程的函数如下：

```
function y = dteq(b,a,x,y0)
N = length(a) - 1; M = length(b) - 1;
lx = length(x);
x = [zeros(1,M),x];
y = [y0,zeros(1,lx)];
aa = a(2:N+1)/a(1);     % aa = [a1 ... aN]/a0
bb = b/a(1);            % bb = [b0 ... bM]/a0
for n = N+1:N+lx
    y(n) = -aa*y(n-1:-1:n-N)'+bb*x(n-N+M:-1:n-N)';
end
y = y(N+1:N+lx);
% End of function dteq.
```

其中，函数变元 a 和 b 表示系统的有关系数向量 $\boldsymbol{a}$，$\boldsymbol{b}$，$\boldsymbol{a}=[a_0,a_1,\cdots,a_N]$，$\boldsymbol{b}=[b_0,b_1,\cdots,b_M]$；y0 表示初始状态向量，即 $[y[-N],y[-N+1],\cdots,y[-1]]$；x 表示输入序列，即 $[x[0],x[1],\cdots]$；y 表示输出序列，即 $[y[0],y[1],\cdots]$；输出序列的输出长度等于输入序列的长度。

例 10-4　单位样值响应

已知二阶离散时间系统的差分方程

$$y[n]-1.5y[n-1]+y[n-2]=2x[n-1]$$

用递推方法求系统的单位样值响应，设 n 取至 20。

解　M 文件如下：

```
a = [1, -1.5,1];
b = [0,2];
```

```
n = 0 : 20;
x = [1,zeros(1,20)];
y0 = [0,0];
y = dteq(b,a,x,y0);
stem(n,y,'filled')
line([0 20],[0 0])
xlabel('n')
ylabel('y[n]')
```

输出波形如图 10－5 所示，$y[n]$的部分结果如下：

$$y[n]=\{\underset{\underset{n=0}{\uparrow}}{0},2,3,2.5,0.75,-1.375,-2.8125,\cdots\}$$

MATLAB 提供了求解单位样值响应的指令

```
[h,n] = impz(b,a)
[h,n] = impz(b,a,N)
```

其中，N 为样点数目；n 为序号列向量；h 为存储单位样值响应数据的列向量。

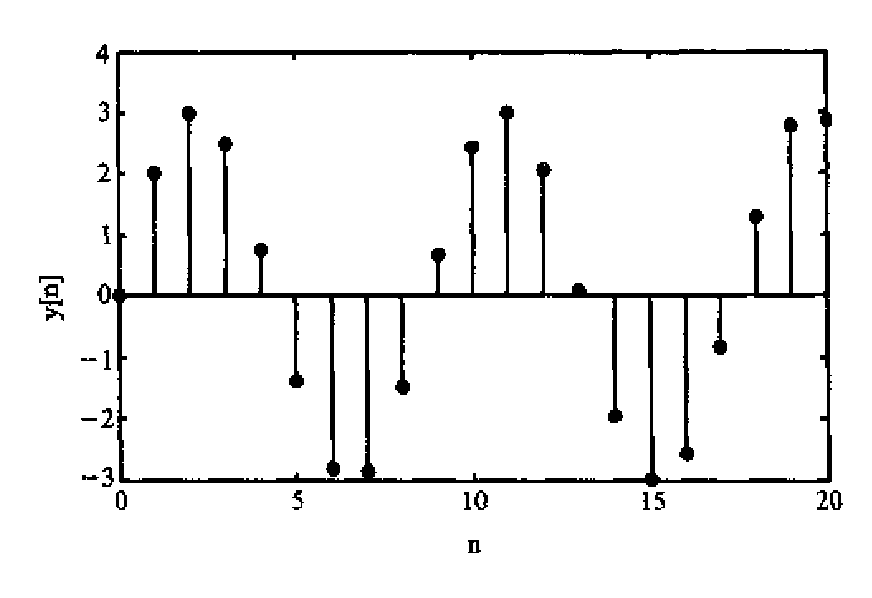


图 10－5 例 10－4 的输出波形

例 10－5 差分方程的递推求解

差分方程如例 10－4 所示，若 $y[-2]=2$，$y[-1]=1$，$x[n]$为单位阶跃序列$\varepsilon[n]$，求 $y[n]$。

解 M 文件如下：

```
a = [1, -1.5,1];
b = [0,2];
```

```
n = 0:20;
x = ones(1,length(n));
y0 = [2,1];
y = dteq(b,a,x,y0);
stem(n,y,'filled')
line([0 20],[0 0])
xlabel('n')
ylabel('y[n]')
```

输出波形如图 10－6 所示。

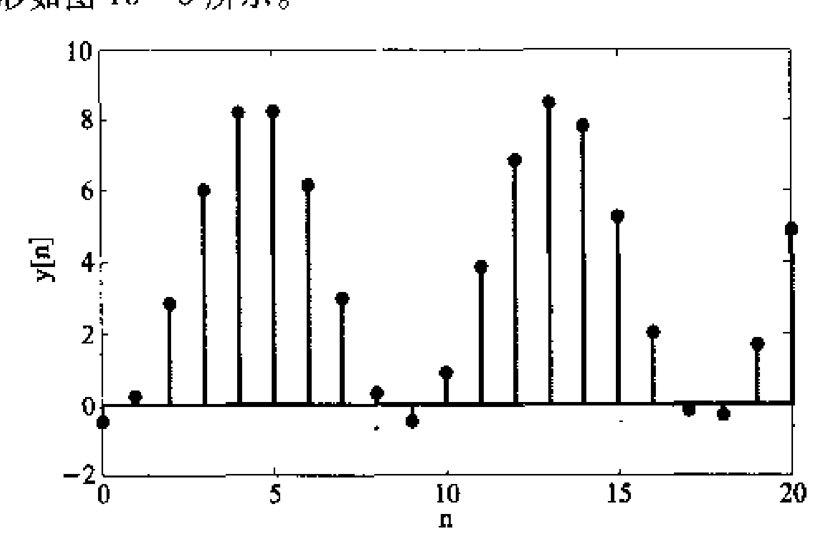


图 10－6 例 10－5 的输出波形

如果只求系统的零状态响应，也可直接使用 MATLAB 提供的指令 filter，其格式为

```
y = filter(b,a,x)
```

其中，x 为输入信号的行向量。例如，若使用该指令求解例 10－4 的单位阶跃响应，M 文件如下：

```
a = [1,-1.5,1];
b = [0,2];
n = 0:20;
x = ones(1,length(n));
y = filter(b,a,x);
stem(n,y)
xlabel('n')
ylabel('y[n]')
```

10.3 卷积求和

卷积求和公式定义为

$$y[n]=h[n]*x[n]=\sum_{m=-\infty}^{\infty}h[m]x[n-m] \qquad (10-6)$$

如果 $h[n]$ 为系统的单位样值响应，$x[n]$ 为输入，则 $y[n]$ 就为系统的输出。假设 $h[n]$ 和 $x[n]$ 都是有限长序列，区间分别为

$$n_{h1}\leqslant n\leqslant n_{h2}$$

$$n_{x1}\leqslant n\leqslant n_{x2}$$

那么，$y[n]$ 的区间一定为

$$n_{h1}+n_{x1}\leqslant n\leqslant n_{h2}+n_{x2} \qquad (10-7)$$

计算有限长序列卷积的指令格式如下：

```
y = conv(h,x)
```

其中，h 和 x 是存储 $h[n]$ 和 $x[n]$ 的行向量。然而，指令 conv 并不给出 $y[n]$ 的区间，而这个区间是有意义的，因此，在求出 $y[n]$ 样本的同时，还必须给出对应的区间。

例 10-6 卷积求和

设 $h[n]$ 和 $x[n]$ 分别为

$$h[n]=\begin{cases}1 & 0\leqslant n\leqslant 5\\ 0 & \text{其余 } n\end{cases}$$

$$x[n]=\begin{cases}n & -2\leqslant n\leqslant 5\\ 0 & \text{其余 } n\end{cases}$$

求 $h[n]$ 和 $x[n]$ 的卷积。

解 M 文件如下：

```
nh = 0:5; Nh = length(nh);
h = ones(1,Nh);
nx = -2:5; Nx = length(nx);
x = nx;
ny = (nh(1) + nx(1)):(nh(Nh) + nx(Nx));
y = conv(h,x);
stem(ny,y,'filled');
line([-2 10],[0 0])
```

```
xlabel('n'); ylabel('y[n]');
```

输出图形如图 10-7 所示。

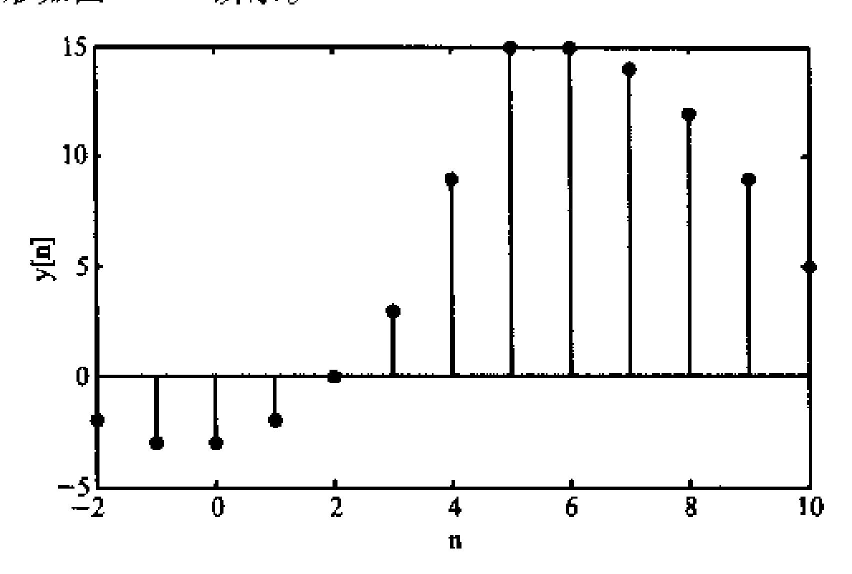


图 10-7 例 10-6 的输出图形

如果采用相同的横坐标绘制 $h[n]$、$x[n]$ 和 $y[n]$ 的图形，在绘图之前，还需要确定序号的最大范围，并给各序列补上相应的零元素。求解两个序列的卷积并绘制图形的函数如下：

```
function [ny,y] = convstem(nh,h,nx,x)
% CONVSTEM   convlution of sequences h and x.
%
Nh = length(nh);
Nx = length(nx);
ny = (nh(1) + nx(1)):(nh(Nh) + nx(Nx)); Ny = length(ny);
y = conv(h,x);
n1 = min([nx(1),nh(1),ny(1)]);
n2 = max([nx(Nx),nh(Nh),ny(Ny)]);
n = n1:n2;
xx = [zeros(1,nx(1) - n1),x,zeros(1,n2 - nx(Nx))];
subplot(3,1,1)
stem(n,xx,'filled')
line([-2 10],[0 0])
ylabel('sequence 1')
```

```
hh = [zeros(1,nh(1) - n1),h,zeros(1,n2 - nh(Nh))];
subplot(3,1,2)
stem(n,hh,'filled')
line([-2 10],[0 0])
ylabel('sequence 2')
yy = [zeros(1,ny(1) - n1),y,zeros(1,n2 - ny(Ny))];
subplot(3,1,3)
stem(n,yy,'filled')
line([-2 10],[0 0])
ylabel('output')
xlabel('n')
% End of function convstem
```

对例10-6,如果用convstem求解,M文件如下：

```
nh = 0:5; Nh = length(nh);
h = ones(1,Nh);
nx = -2:5; Nx = length(nx);
x = nx;
convstem(nh,h,nx,x);
```

输出图形如图10-8所示。

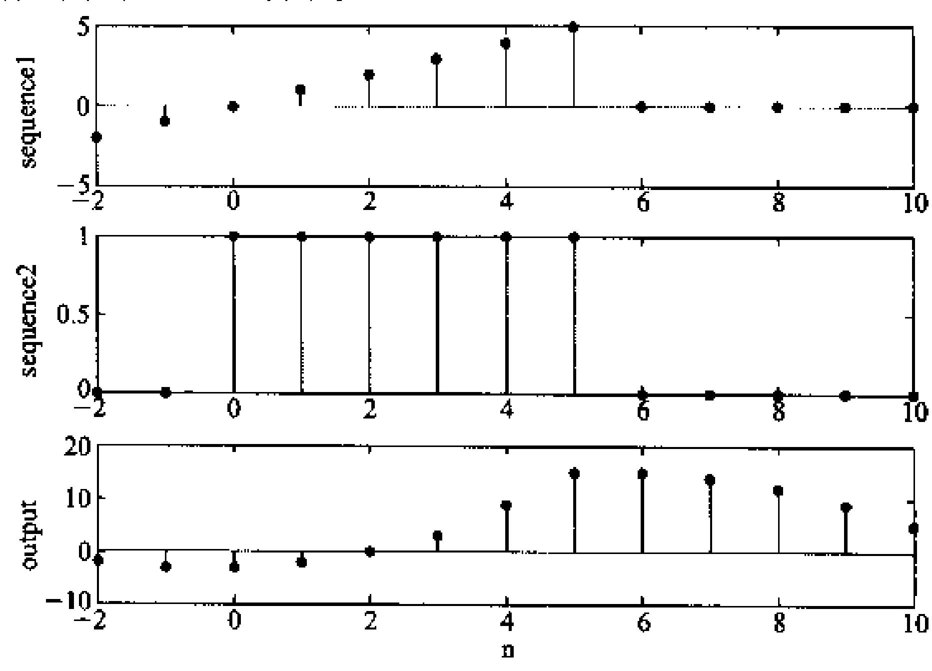


图10-8 例10-6各序列的图形

10.4 数值卷积

连续时间信号 $h(t)$ 与 $x(t)$ 的卷积积分

$$y(t)=h(t)*x(t)=\int_{-\infty}^{\infty}h(\tau)x(t-\tau)\mathrm{d}\tau \tag{10-8}$$

设按等间隔 T 对信号取样，式(10-8)有以下近似关系：

$$y(nT)\approx\sum_{k=-\infty}^{\infty}Th(kT)x(nT-kT) \tag{10-9}$$

若令 $h[n]=h(nT)$，$x[n]=x(nT)$，$y[n]=y(nT)$，式(10-9)可表示成

$$y[n]=\sum_{k=-\infty}^{\infty}Th[k]x[n-k]=Th[n]*x[n] \tag{10-10}$$

其中，$h[n]*x[n]$ 表示序列 $h[n]$ 与 $x[n]$ 的卷积求和。如果式(10-8)中 $h(t)$ 表示连续时间系统的单位冲激响应，由式(10-9)，则对应的离散时间系统的单位样值响应就为 $Th[n]$，即取样间隔乘以 $h(t)$ 的样本。

例 10-7　数值卷积

已知一连续时间系统的单位冲激响应

$$h(t)=\mathrm{e}^{-5t}\varepsilon(t)$$

输入信号 $x(t)$ 为矩形脉冲，如图 10-9 所示，求零状态响应 $y(t)$。

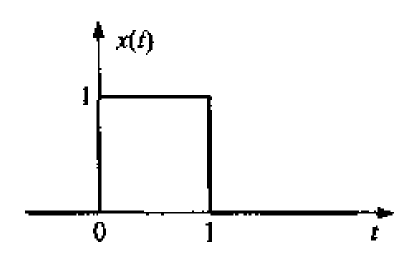


图 10-9　矩形脉冲

解　M 文件如下：

```
ts =0.01;
n =0:200; t = n * ts;
h = exp( -5 * t) ; h(1) =0.5;
h = h * ts;
x = [ ones(1,100),0.5,zeros(1,100) ];
y = conv(h,x);
y = y(1:length(n));
```

```
plot(t,y)
xlabel('t')
ylabel('y(t)')
grid
```

输出波形如图10－10所示。

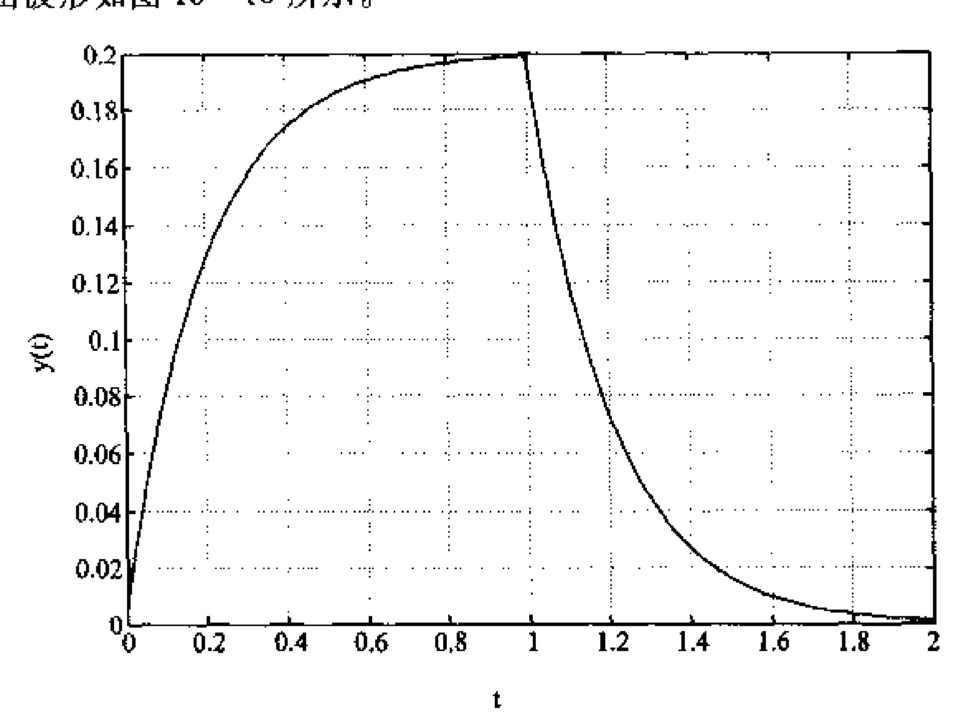


图10－10 例10－7的输出波形

10.5 系统的状态空间分析

在对系统建立数学方程时，如果把系统内部的某些信号作为变量，线性连续时间系统的方程可写作

$$\boldsymbol{x}'(t)=\boldsymbol{A}\boldsymbol{x}(t)+\boldsymbol{B}u(t) \qquad (10-11a)$$

$$\boldsymbol{y}(t)=\boldsymbol{C}\boldsymbol{x}(t)+\boldsymbol{D}u(t) \qquad (10-11b)$$

式(10－11a)称为系统的状态方程，$u(t)$为输入，$\boldsymbol{x}(t)$为状态向量[①]；式(10－11b)称为状态空间输出方程，$\boldsymbol{y}(t)$为输出向量。

根据式(10－11)求解系统响应的指令如下：

① 在I/O方程中，$x(t)$为输入，在状态方程中，$\boldsymbol{x}(t)$一般用于表示状态向量。

(1) 零输入响应。

```
initial(sys,x0)
initial(sys,x0,t)
[y,t,x] = initial(sys,x0)
```

其中,sys 表示系统,它可用状态空间方程定义,格式为

```
sys = ss(A,B,C,D)
```

x0 为状态空间初始条件 $\boldsymbol{x}(0)$;t 为计算的终止时间或时间行向量;对该函数指令,如果输出变元不给出,initial 函数会在屏幕上绘制出系统的零输入响应曲线,如果输出变元给出,则不绘制曲线,只给出有关量的计算数据。

(2) 冲激响应。

```
impulse(sys)
impulse(sys,t)
[y,t,x] = impulse(sys)
```

该指令只能用于输出信号不含冲激函数的系统,即状态空间模型中系数矩阵 $\boldsymbol{D}$ 必须为零矩阵。

(3) 阶跃响应。

```
step(sys)
step(sys,t)
[y,t,x] = step(sys)
```

(4) 任意输入的零状态响应或全响应。

```
lsim(sys,u,t)
lsim(sys,u,t,x0)
[y,t,x] = lsim(sys,u,t,x0)
```

其中,u 为输入信号的样本。当初始条件 x0 不给出时,该指令计算系统的零状态响应。

例 10-8 电路的状态空间分析

电路如图 10-11 所示,已知 $R_1=1\ \text{k}\Omega, R_2=2\ \text{k}\Omega, R_3=3\ \text{k}\Omega, C_1=C_2=C_3=1\ \mu\text{F}, v=10\,\varepsilon(t)\ \text{V}, v_1(0_-)=0\ \text{V}, v_2(0_-)=6\ \text{V}$,求 $t>0$ 时的 $v_2(t)$。

解 利用结点法建立电路的状态方程。对结点 1 和结点 2 应用 KCL,有

结点 1: $$C_1v_1'+C_2(v_1'-v_2')=\frac{1}{R_1}(v-v_1)+\frac{1}{R_2}(v_2-v_1)$$

结点 2: $$C_3v_2'+C_2(v_2'-v_1')=\frac{1}{R_2}(v_1-v_2)-\frac{1}{R_3}v_2$$

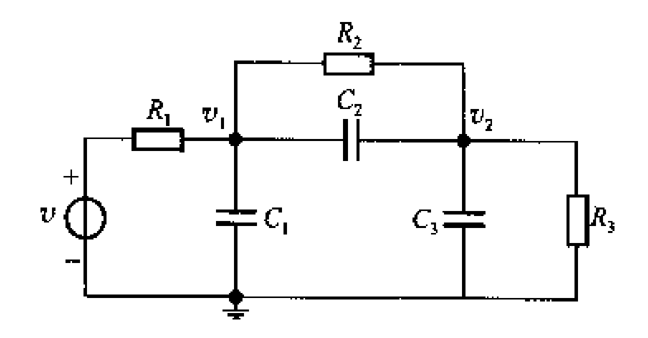


图 10-11　例 10-8 电路

即

$$\begin{bmatrix} C_1+C_2 & -C_2 \\ -C_2 & C_3+C_2 \end{bmatrix}\begin{bmatrix} v_1' \\ v_2' \end{bmatrix}=\begin{bmatrix} -\frac{1}{R_1}-\frac{1}{R_2} & \frac{1}{R_2} \\ \frac{1}{R_2} & -\frac{1}{R_3}-\frac{1}{R_2} \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}+\begin{bmatrix} \frac{1}{R_1} \\ 0 \end{bmatrix}v$$

则状态空间方程为

$$\begin{bmatrix} v_1' \\ v_2' \end{bmatrix}=\begin{bmatrix} C_1+C_2 & -C_2 \\ -C_2 & C_3+C_2 \end{bmatrix}^{-1}\begin{bmatrix} -\frac{1}{R_1}-\frac{1}{R_2} & \frac{1}{R_2} \\ \frac{1}{R_2} & -\frac{1}{R_3}-\frac{1}{R_2} \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}+\begin{bmatrix} C_1+C_2 & -C_2 \\ -C_2 & C_3+C_2 \end{bmatrix}^{-1}\begin{bmatrix} \frac{1}{R_1} \\ 0 \end{bmatrix}v$$

$$v_2=[0,1]\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

M 文件如下：

```
g1 = 1/1e3; g2 = 1/2e3; g3 = 1/3e3; c = 1e-6;
M = c * [2, -1; -1,2];
Minv = inv(M);
A = Minv * [ -g1 - g2,g2;g2, -g2 - g3];
B = Minv * [g1;0];
C = [0,1];D = [0];
x0 = [0;6];
sys = ss(A,B,C,D);
t = 0:1e-4:0.02;
u = 10 * ones(1,length(t));
lsim(sys,u,t,x0);
grid
```

输出波形如图 10－12 所示。

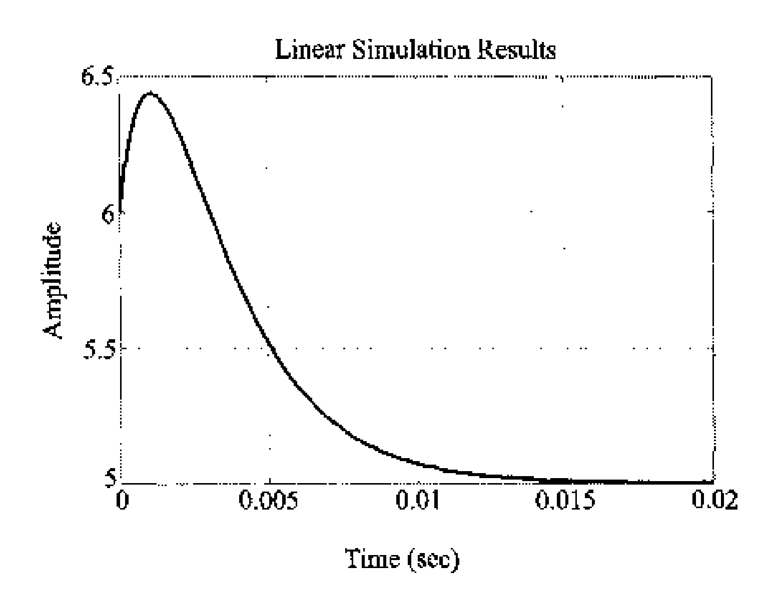


图 10－12　例 10－8 的输出波形

系统的零输入响应采用 initial 或 lsim 求解，但只能用于状态空间方程，如果给定的是输入输出方程，可先对其转换。在零输入条件下，有

$$a_0\frac{\mathrm{d}^N y(t)}{\mathrm{d}t^N}+a_1\frac{\mathrm{d}^{N-1}y(t)}{\mathrm{d}t^{N-1}}+\cdots+a_N y(t)=0$$

若令 $x_1=\dfrac{\mathrm{d}^{N-1}y(t)}{\mathrm{d}t^{N-1}},x_2=\dfrac{\mathrm{d}^{N-2}y(t)}{\mathrm{d}t^{N-2}},\cdots,x_N=y$，则有

$$\begin{cases}x_1'=-\dfrac{1}{a_0}(a_1x_1+a_2x_2+\cdots+a_Nx_N)\\ x_2'=x_1\\ \vdots\\ x_N'=x_{N-1}\end{cases}$$

于是，状态空间方程为（以 5 阶系统为例）

$$\begin{bmatrix}x_1'\\x_2'\\x_3'\\x_4'\\x_5'\end{bmatrix}=\begin{bmatrix}-a_1/a_0 & -a_2/a_0 & -a_3/a_0 & -a_4/a_0 & -a_5/a_0\\ 1&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\end{bmatrix}\tag{10-12}$$

$$y=[0,0,0,0,1]\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\end{bmatrix} \tag{10-13}$$

初始条件

$$\begin{bmatrix}x_1(0)\\x_2(0)\\x_3(0)\\x_4(0)\\x_5(0)\end{bmatrix}=\begin{bmatrix}y^{(4)}(0)\\y^{(3)}(0)\\y^{(2)}(0)\\y^{(1)}(0)\\y(0)\end{bmatrix} \tag{10-14}$$

按式(10-12)~(10-14)所示的状态方程则可分析系统的零输入响应,在10.1节给出的 initialr 函数按该方法编写。

离散时间系统也可在状态空间描述,方程如下:

$$\boldsymbol{x}[n+1]=\boldsymbol{A}\boldsymbol{x}[n]+\boldsymbol{B}u[n] \tag{10-15a}$$

$$\boldsymbol{y}[n]=\boldsymbol{C}\boldsymbol{x}[n]+\boldsymbol{D}u[n] \tag{10-15b}$$

其中,$\boldsymbol{x}[n]$为状态向量;$u[n]$为输入;$\boldsymbol{y}[n]$为输出向量。

用递推方法求状态空间方程的函数如下:

```
function [y,w] = dtss(A,B,C,D,u,x0)
%
x(1,:) = x0;
for n = 2:length(u)
    x(n,:) = (A * x(n-1,:)' + B * u(n-1))';
end
for n = 1:length(u)
    y(n) = C * x(n,:)' + D * u(n);
end
% End of function dtss.
```

其中,输入变元 x0 为初始状态的行向量;输出变元 x 的行对应于序号,列对应于状态变量。

例 10-9 离散时间状态空间方程的求解

已知

$$\begin{bmatrix} x_1[n+1] \\ x_2[n+1] \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u[n]$$

$$y[n] = \begin{bmatrix} 3 & 0 \end{bmatrix} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix} + 2u[n]$$

$$u[n] = n+1, \quad \begin{bmatrix} x_1[0] \\ x_2[0] \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

求 $y[n]$,设 $0 \leqslant n \leqslant 5$。

解 M 文件如下：

```
A = [0,2;1,0]; B = [1;0];
C = [3,0]; D = 2;
n = 0:5;
u = n + 1;
x0 = [0,0];
y = dtss(A,B,C,D,u,x0)
```

执行程序后得

$$y[n] = \{\underset{n=0}{\underset{\uparrow}{2}}, 7, 12, 23, 34, 57, \cdots\}$$

第十一章　拉普拉斯变换

分析连续时间系统的响应、稳定性、频率响应，拉普拉斯变换起着非常重要的作用，它把系统的微分方程变换为代数方程，从而简化了计算。尽管如此，由于复频率 s 并非一个具体的数值，高阶系统的手工分析还是具有比较大的计算量。因此，计算机辅助复频域分析就十分必要。本章首先介绍求解拉普拉斯变换的符号方法和部分分式法，其次介绍系统的几种描述方法及它们之间的转换指令，最后介绍电路分析函数 sana 的复频域分析功能。

11.1　拉普拉斯变换

本节介绍正、逆拉普拉斯变换的符号求解指令，以及 s 域有理函数的部分分式展开指令。

信号 $x(t)$ 的单边拉普拉斯变换定义为

$$X(s) = \int_{0_-}^{\infty} x(t)e^{-st}\mathrm{d}t \tag{11-1}$$

与上式对应的指令为

```
xs = laplace(xt,t,s)
```

其中，xt 为 $x(t)$ 的符号表达式；t 为积分变量 t；s 为复频率 s；xs 为 $x(t)$ 的拉普拉斯变换 $X(s)$。如果 xt 中 t 为 MATLAB 规定的积分变量，而且用 s 表示复频率，上面的指令也可简写成

```
xs = laplace(xt)
```

为了避免出错，最好采用完整的指令格式。

例 11－1　正弦函数与指数函数的拉普拉斯变换

求 $x_1(t) = \sin(2t)\varepsilon(t)$ 和 $x_2(t) = e^{-at}\varepsilon(t)$ 的拉普拉斯变换。

解　M 文件如下：

```
syms t a;       % 指定 t 和 a 为符号变量
x1t = sin(2 * t);
```

```
x2t = exp( - a * t);
x1s = laplace(x1t)
x2s = laplace(x2t,t,s)
```

输出结果：

```
    x1s = 2/(s^2 +4)
    x2s = 1/(s + a)
```

例 11 -2 冲激函数与阶跃函数的拉普拉斯变换

求 $\delta(t)$ 和 $\varepsilon(t-a)$ 的拉普拉斯变换，其中 $a>0$。

解 冲激函数 $\delta(t)$ 和阶跃函数 $\varepsilon(t)$ 在符号分析程序 Maple 中分别用 Dirac(t) 和 Heaviside(t) 表示，高阶冲激函数 $\delta^{(n)}(t)$ 用 Dirac(n,t) 表示，由于 MATLAB 本身对冲激函数和阶跃函数没有定义，因此必须把它们定义为符号对象。

```
syms t;
syms a positive;    % 指定 a 为取正值的符号变量
dt = sym('Dirac(t)');
et = sym('Heaviside(t - a)');
ds = laplace(dt,t,s)
es = laplace(et,t,s)
```

输出结果：

```
ds = 1
es = exp( -a * s)/s
```

逆拉普拉斯变换式为

$$x(t)=\frac{1}{2\pi j}\int_{\alpha-j\infty}^{\alpha+j\infty}X(s)e^{st}ds \tag{11-2}$$

实现上式运算的指令格式为

```
    xt = ilaplace(xs,s,t)
    xt = ilaplace(xs)
```

如果 s 是 xs 符号表达式的隐含自变量，t 是时间变量，则 s、t 可省略。注意：式(11 -2)求出的 $x(t)$ 只适合于 $t>0_-$。

例 11 -3 符号逆拉普拉斯变换

求下列复频域函数的逆拉普拉斯变换。

(1) $X_1(s)=\arctan\left(\frac{1}{s}\right)$

(2) $X_2(s)=\frac{s^2}{s^2+3s+2}$

解 M 文件如下：

```
syms s;
x1s = atan(1/s);   % atan 为反正切函数
x2s = s^2/(s^2 +3 * s +2);
x1t = ilaplace(x1s)
x2t = ilaplace(x2s)
```

输出结果为

```
x1t = sin(t)/t
x2t = Dirac(t) -4 * exp( -2 * t) + exp( -t)
```

则

$$x_1(t) = \frac{\sin(t)}{t}\varepsilon(t)$$

$$x_2(t) = \delta(t) + (-4e^{-2t} + e^{-t})\varepsilon(t)$$

例 11-4 有理分式的逆拉普拉斯变换

求 $X(s) = \dfrac{-2s^2+7s+19}{s^3+5s^2+17s+13}$的逆拉普拉斯变换。

解 由于 $X(s)$为有理分式，可先将分子、分母多项式的有关系数用数组表示，再利用 poly2sym 函数将其转化为多项式。

```
syms s;
b = [ -2,7,19];
a = [1,5,17,13];
xs = poly2sym(b,s)/poly2sym(a,s);
xt = ilaplace(xs)
```

输出结果为

```
xt = exp( -t) -3 * exp( -2 * t) * cos(3 * t) +4 * exp( -2 * t) * sin(3 * t)
```

例 11-5 时移性质和微分性质

设 $x(t) \leftrightarrow X(s)$，验证拉普拉斯变换的时移性质和微分性质：

$$x(t-t_0)\varepsilon(t-t_0) \leftrightarrow e^{-st_0}X(s)$$

$$\frac{dx(t)}{dt} \leftrightarrow sX(s) - x(0_-)$$

证

```
syms t s;syms t0 positive;
x1t = sym('x(t - t0)') * sym('Heaviside(t - t0)')
```

```
x1s = laplace(x1t,t,s)
x1t = ilaplace(x1s,s,t)
x2t = diff(sym('x(t)'),t)
x2s = laplace(x2t,t,s)
x2t = ilaplace(x2s,s,t)
```

输出结果为

```
x1t  =  x(t - t0) * Heaviside(t - t0)
x1s  =  exp( - s * t0) * laplace(x(t),t,s)
x1t  =  x(t - t0) * Heaviside(t - t0)
x2t  =  diff(x(t),t)
x2s  =  s * laplace(x(t),t,s) - x(0)
x2t  =  diff(x(t),t)
```

当 $X(s)$ 为有理分式时，逆拉普拉斯变换常用部分分式法求解。设

$$X(s)=\frac{B(s)}{A(s)}=\frac{b_0s^M+b_1s^{M-1}+\cdots+b_{M-1}s+b_M}{a_0s^N+a_1s^{N-1}+\cdots+a_{N-1}s+a_N} \tag{11-3a}$$

如果所有极点互不相等，$X(s)$ 可展开为

$$X(s)=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}+\cdots+\frac{r_N}{s-p_N}+k_1s^{M-N}+\cdots+k_{M-N}s+k_{M-N+1} \tag{11-3b}$$

如果 p_m 为 q 阶极点，与其对应的展开项为

$$\frac{r_m}{s-p_m}+\frac{r_{m+1}}{(s-p_m)^2}+\cdots+\frac{r_{m+q-1}}{(s-p_m)^q} \tag{11-3c}$$

对式(11-3a)进行部分分式展开的指令为

```
[r,p,k] = residue(b,a)
```

其中，b 是由 $X(s)$ 分子多项式系数组成的行向量 $\boldsymbol{b}$，$\boldsymbol{b}=[b_0,b_1,\cdots,b_M]$，注意 $\boldsymbol{b}$ 中元素以 s 的降幂顺序排列各个系数，当多项式中某项空缺时其系数为 0；a 是由 $X(s)$ 分母多项式系数组成的行向量 $\boldsymbol{a}$，$\boldsymbol{a}=[a_0,a_1,\cdots,a_N]$；r 是留数列向量 $\boldsymbol{r}$，$\boldsymbol{r}=[r_1,r_2,\cdots,r_N]^{\mathrm{T}}$；p 是极点列向量 $\boldsymbol{p}$，$\boldsymbol{p}=[p_1,p_2,\cdots,p_N]^{\mathrm{T}}$；k 是直接项系数行向量 $\boldsymbol{k}$。

例 11-6　有理分式的部分分式展开

求 $X(s)=\dfrac{-2s^2+7s+19}{s^3+5s^2+17s+13}$ 的逆拉普拉斯变换。

解　使用 residue 函数求解，M 文件如下：

```
b = [ -2,7,19];
a = [1,5,17,13];
[r,p,k] = residue(b,a)
```

输出结果为

```
r =
   -1.5000 - 2.0000i
   -1.5000 + 2.0000i
   1.0000
p =
   -2.0000 + 3.0000i
   -2.0000 - 3.0000i
   -1.0000
k = []
```

$X(s)$ 的展开式为

$$X(s)=\frac{-1.5-\mathrm{j}2}{s+2-\mathrm{j}3}+\frac{-1.5+\mathrm{j}2}{s+2+\mathrm{j}3}+\frac{1}{s+1}$$

于是

$$\begin{aligned}x(t)&=(-1.5-\mathrm{j}2)\mathrm{e}^{(-2+\mathrm{j}3)t}+(-1.5+\mathrm{j}2)\mathrm{e}^{(-2-\mathrm{j}3)t}+\mathrm{e}^{-t}\\&=-1.5\mathrm{e}^{-2t}(\mathrm{e}^{\mathrm{j}3t}+\mathrm{e}^{-\mathrm{j}3t})-\mathrm{j}2\mathrm{e}^{-2t}(\mathrm{e}^{\mathrm{j}3t}-\mathrm{e}^{-\mathrm{j}3t})+\mathrm{e}^{-t}\\&=-3\mathrm{e}^{-2t}\cos(3t)+4\mathrm{e}^{-2t}\sin(3t)+\mathrm{e}^{-t}\end{aligned}$$

或

$$\begin{aligned}x(t)&=2\mathrm{Re}[(-1.5-\mathrm{j}2)\mathrm{e}^{(-2+\mathrm{j}3)t}]+\mathrm{e}^{-t}\\&=2\mathrm{e}^{-2t}\mathrm{Re}[-1.5\mathrm{e}^{\mathrm{j}3t}-\mathrm{j}2\mathrm{e}^{\mathrm{j}3t}]+\mathrm{e}^{-t}\\&=\mathrm{e}^{-2t}[-3\cos(3t)+4\sin(3t)]+\mathrm{e}^{-t}\end{aligned}$$

函数 residue 也可将部分分式转化为两个多项式之比形式，其格式为

```
[b,a] = residue(r,p,k)
```

对上例中的部分分式进行转化，文件如下：

```
r = [ -1.5 - 2j; -1.5 + 2j;1];
p = [ -2 + 3j; -2 - 3j; -1];
k = [];
```

```
[b,a] = residue(r,p,k)
```

输出结果为

```
b =   -2    7    19
a =    1    5    17    13
```

例 11-7 有理函数含有重极点时的部分分式展开

利用部分分式法求解 $X(s)=\dfrac{3s^4+6}{(s+1)^3(s-2)}$ 的逆拉普拉斯变换。

解 $X(s)$ 的分母多项式在 $p_1=-1$ 处具有 3 重根,用根构造多项式的指令为

```
poly(r)
```

其中,r 为多项式的根向量。M 文件如下:

```
p = [-1, -1, -1, 2];   % 极点行向量
a = poly(p)    % 构造分母多项式
b = [3,0,0,0,6];
[r,p,k] = residue(b,a)
```

输出结果为

```
a =
    1    1    -3    -5    -2
r =
    2.0000
   -5.0000
    3.0000
   -3.0000
p =
    2.0000
   -1.0000
   -1.0000
   -1.0000
k = 3
```

则(注意留数与极点的对应关系)

$$X(s)=\frac{2}{s-2}+\frac{-5}{(s+1)}+\frac{3}{(s+1)^2}+\frac{-3}{(s+1)^3}+3$$

于是,$X(s)$的逆拉普拉斯变换

$$x(t)=2e^{2t}+\left(-5+3t-\frac{3}{2}t^2\right)e^{-t}+3\delta(t)$$

在利用指令 residue 对有理分式 $X(s)$ 展开时,当 $X(s)$ 为假分式,即分子多项式的次数 M 大于或等于分母多项式的次数 N 时,$X(s)$能够表示成 s 的多项式与真分式的和,即

$$X(s)=Q(s)+\frac{R(s)}{A(s)} \tag{11-4}$$

其中,$Q(s)$为 s 的多项式,次数等于 $M-N$;$R(s)$的次数小于 $A(s)$的次数。执行上式运算的指令为

```
[q,r] = deconv(b,a)
```

其中,b 和 a 分别为 $X(s)$分子、分母多项式的系数向量;q 和 r 分别为 $Q(s)$和 $R(s)$的系数向量。

$Q(s)$和 $R(s)$一旦求出后,$X(s)$的逆拉普拉斯变换为 $Q(s)$的逆变换与 $R(s)/A(s)$的逆变换的和。

当 $X(s)$为有理分式时,指令 impulse 能够直接给出 $x(t)$在 $t\geqslant 0_+$范围的曲线,其调用格式为

```
impulse(b,a)
impulse(b,a,tf)
```

其中,b 和 a 分别为 $X(s)$分子、分母多项式的系数向量;tf 为数值计算的终止时间。指令 impulse 总是从零时刻开始,而且忽略响应中的冲激函数。

例 11-8 $X(s)$的时域曲线

绘制 $X(s)=\dfrac{1}{s^2+3s+2}$的时域波形,计算终止时间取 $t_f=8$ s。

解 M 文件如下:

```
b = 1;
a = [1,3,2];
impulse(b,a,8)
```

$x(t)$的波形如图 11-1 所示。

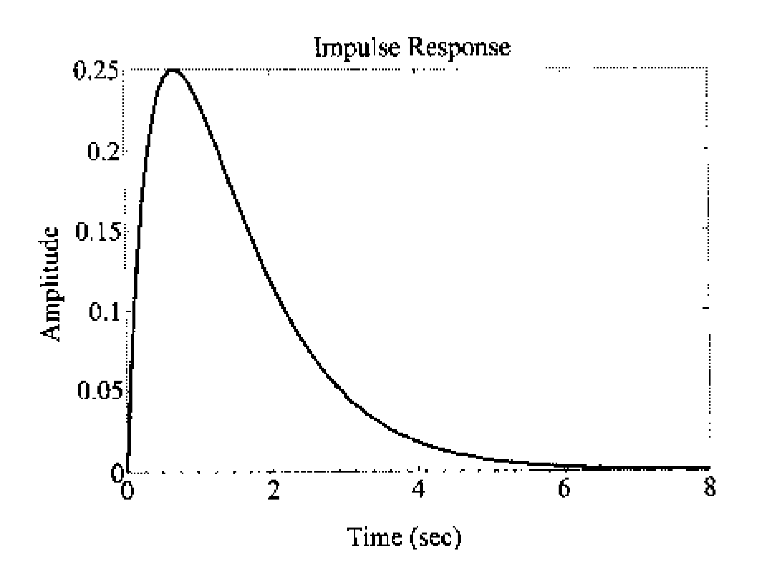


图 11-1 例 11-8 的时域波形

11.2 系统模型

系统模型的常见形式有：传递函数模型、极零点增益模型、状态空间模型等。传递函数模型为

$$H(s)=\frac{B(s)}{A(s)}=\frac{b_0s^M+b_1s^{M-1}+\cdots+b_{M-1}s+b_M}{a_0s^N+a_1s^{N-1}+\cdots+a_{N-1}s+a_N} \tag{11-5}$$

将传递函数的分子、分母作因式分解，可得系统的极零点增益模型为

$$H(s)=k\frac{(s-z_1)(s-z_2)\cdots(s-z_M)}{(s-p_1)(s-p_2)\cdots(s-p_N)} \tag{11-6}$$

系统的状态空间模型为

$$\boldsymbol{x}'=\boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}\boldsymbol{u} \tag{11-7}$$

$$\boldsymbol{y}=\boldsymbol{C}\boldsymbol{x}+\boldsymbol{D}\boldsymbol{u} \tag{11-8}$$

其中，$\boldsymbol{x}$ 为状态向量；$\boldsymbol{u}$ 为输入；$\boldsymbol{y}$ 为输出；$\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$、$\boldsymbol{D}$ 是方程的系数矩阵。对单输入单输出系统，由状态方程和输出方程可求出系统的传递函数

$$\boldsymbol{H}(s)=\boldsymbol{C}(s\boldsymbol{1}-\boldsymbol{A})^{-1}\boldsymbol{B}+\boldsymbol{D} \tag{11-9}$$

其中，1 是单位矩阵。

在 MATLAB 的控制系统工具箱中，系统可用以上任一种模型定义。传递函数模型的语句为

```
sys = tf(b,a)
```

其中,b 为由传递函数分子多项式系数组成的向量,按 s 的降幂排列;a 为由传递函数分母多项式系数组成的向量;sys 为存储传递函数数据的传递函数。

例如,输入语句为

```
b = [2, -1];
a = [1,3,2];
sys = tf(b,a)
```

输出结果为

```
Transfer function:
   2 s - 1
--------------------
s^2 + 3 s + 2
```

极零点增益模型语句为

```
sys = zpk(z,p,k)
```

其中,z 为存放零点数据的列向量;p 为存放极点数据的列向量;k 为增益。若键入

```
z = 0.5;
p = [ -1; -2];
k = 2;
sys = zpk(z,p,k)
```

输出结果为

```
Zero/pole/gain:
 2 (s-0.5)
--------------------
(s+1) (s+2)
```

状态空间模型语句为

```
sys = ss(A,B,C,D)
```

其中,A、B、C、D 为式(11-7)和式(11-8)中的有关系数矩阵。

例如:

```
a = [ -1,1.225;0, -2];
b = [0;1.414];
c = [ -1.732,1.414];
d = 0;
sys = ss(a,b,c,d)
```

输出结果为

```
a =
            x1      x2
   x1       -1   1.225
   x2        0      -2
b =
            u1
   x1        0
   x2    1.414
c =
             x1        x2
   y1    -1.732     1.414
d =
         u1
   y1     0
```

以上三种模型之间可以相互转换,指令为

```
tfsys = tf(sys)
zpksys = zpk(sys)
sssys = ss(sys)
```

其中,sys 表示系统模型;tf(sys)转换为传递函数形式;zpk(sys)转换为极零点增益形式;ss(sys)转换为状态空间形式。

例 11-9 系统模型的转换

已知系统的状态空间方程

$$\begin{bmatrix} x_1'(t) \\ x_2'(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -3 & -2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [2,1] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

求传递函数和极零点。

解 根据已知的状态空间模型,利用指令 tf(sys)和 zpk(sys)求解。

```
a = [0,1; -3, -2];
b = [0;1];
c = [2,1];
d = 0;
```

```
sssys = ss(a,b,c,d);
tfsys = tf(sssys)
zpksys = zpk(sssys)
```

输出结果为

```
Transfer function:
      s + 2
-------------------------
s^2 + 2 s + 3
Zero/pole/gain:
      (s+2)
-------------------------
(s^2 + 2s + 3)
```

对单输入单输出系统，提取传递函数模型、极零点增益模型、状态空间模型有关数据的指令分别为

```
[b,a] = tfdata(sys,'v')
[z,p,k] = zpkdata(sys,'v')
[a,b,c,d] = ssdata(sys)
```

其中，'v'表示输出数据直接以向量形式给出。

例 11-10 系统模型的数据提取

提取例 11-9 所示状态空间方程的各模型数据。

解 M 文件如下：

```
a = [0,1;-3,-2];
b = [0;1];
c = [2,1];
d = 0;
sys = ss(a,b,c,d);
disp('Transfer function data')
[b,a] = tfdata(sys,'v')
disp('zero-pole-gain data')
[z,p,k] = zpkdata(sys,'v')
disp('state-space data')
[a,b,c,d] = ssdata(sys)
```

输出结果为

```
Transfer function data
b =   0     1     2
a =   1.0000    2.0000    3.0000
zero - pole - gain data
z =   -2
p =
   -1.0000 + 1.4142i
   -1.0000 - 1.4142i
k =   1
state - space data
a =
       0      1
      -3     -2
b =
      0
      1
c =   2     1
d =   0
```

11.3 系统函数的极零点图

系统函数的极零点可用多项式求根指令 roots 或指令 tf2zp 求解,其格式为

```
p = roots(a)
z = roots(b)
[z,p,k] = tf2zp(b,a)
```

其中,a 为分母系数向量;p 为极点;b 为分子系数向量;z 为零点; k 为增益。

根据系统模型求极点、零点的指令为

```
p = pole(sys)
z = zero(sys)
```

极零点图用指令 plot 绘制,极点处标注 x,零点位置标注 o。也可直接使用如下指令

```
pzmap(sys)
[p,z] = pzmap(sys)
```

在 s 平面加格线的指令为

```
sgrid
```

例 11－11 极零点图的绘制

绘制系统函数 $H(s)=\dfrac{s+2}{s^2+2s+3}$ 的极零点图。

解 方法 1：

```
b=[1,2];
a=[1,2,3];
zs=roots(b)
ps=roots(a)
plot(real(zs),imag(zs),'o')
hold on
plot(real(ps),imag(ps),'x')
hold on
grid
```

方法 2：

```
b=[1,2];
a=[1,2,3];
[zs,ps,k]=tf2zp(b,a)
plot(real(zs),imag(zs),'o')
hold on
plot(real(ps),imag(ps),'x')
hold on
grid
```

例 11－12 使用 pzmap 指令绘制系统函数的极零点图

绘制系统函数 $H(s)=\dfrac{2s^2+12s+20}{s^3+4s^2+5s+2}$ 的极零点图。

解 M 文件如下：

```
b=[2,12,20];
a=[1,4,5,2];
sys=tf(b,a)
[p,z]=pzmap(sys)
pzmap(sys)
```

```
sgrid
```

输出结果为

```
Transfer function:
  2 s^2 + 12 s + 20
--------------------------------
s^3 + 4 s^2 + 5 s + 2
p =
   -2.0000
   -1.0000 + 0.0000i
   -1.0000 - 0.0000i
z =
   -3.0000 + 1.0000i
   -3.0000 - 1.0000i
```

极零点图如图 11 - 2 所示。

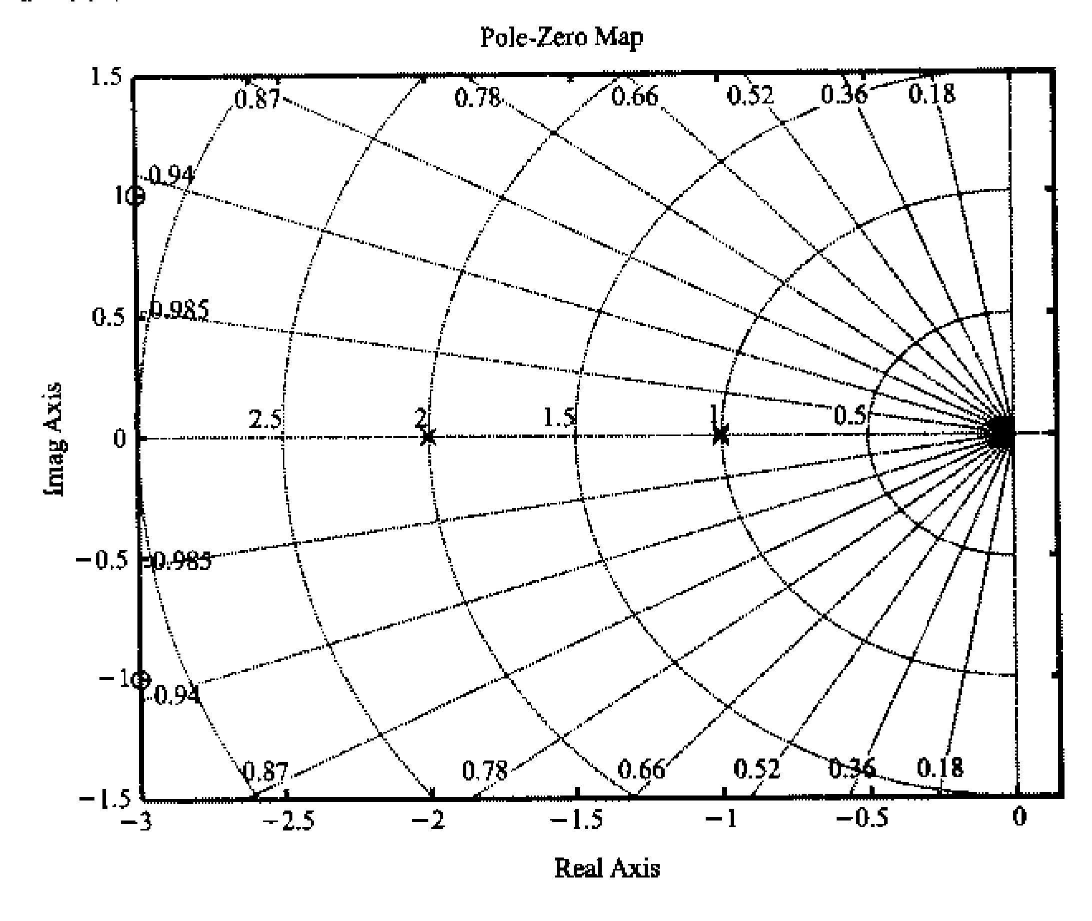


图 11 - 2 极零点图

11.4 频率响应

设系统是稳定的,传递函数为 $H(s)$,输入信号是角频率为 ω_0 的复正弦信号,即 $x(t)=Xe^{j\omega_0 t}(-\infty<t<\infty)$,由卷积公式可求得输出信号

$$y(t)=H(s)\big|_{s=j\omega_0}Xe^{j\omega_0 t}=H(j\omega_0)Xe^{j\omega_0 t} \qquad (11-10)$$

式(11-10)说明输出也是角频率为 ω_0 的复正弦信号。

若用 $|H|$ 和 $\underline{/H}$ 分别表示 $H(j\omega_0)$ 的幅度和相位,即

$$H(j\omega_0)=|H|e^{j\underline{/H}}$$

当输入信号 $x(t)=|X|\cos(\omega_0 t+\varphi_x)(-\infty<t<\infty)$ 时,由式(11-10)得

$$y(t)=|Y|\cos(\omega_0 t+\varphi_y)=|H|\,|X|\cos(\omega_0 t+\varphi_x+\underline{/H})$$

若用 ω 置换 ω_0,$H(j\omega)$ 的幅度 $|H|$ 随 ω 的变化称为系统的幅频响应;$H(j\omega)$ 的相位 $\underline{/H}$ 随 ω 的变化称为系统的相频响应。

例 11-13 频率响应曲线

角频率 ω 在 0.01 rad/s 至 2 rad/s 范围按间隔 0.01 rad/s 取值,绘制

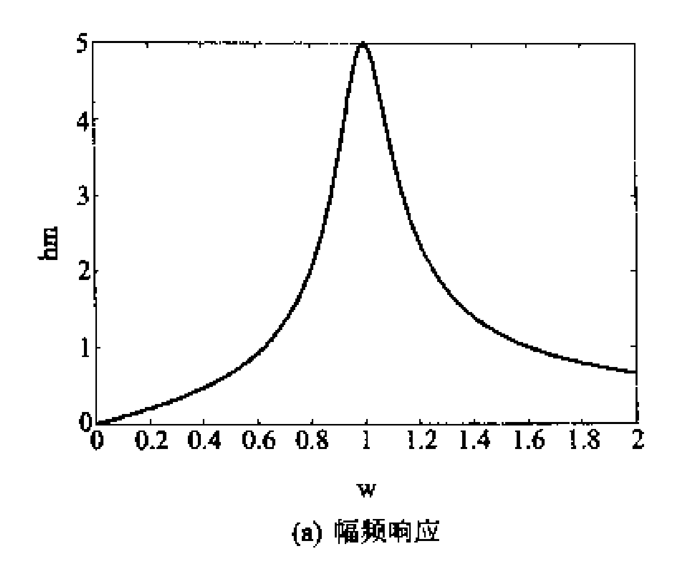
$$H(s)=\frac{s}{s^2+0.2s+1}$$

的频率响应曲线。

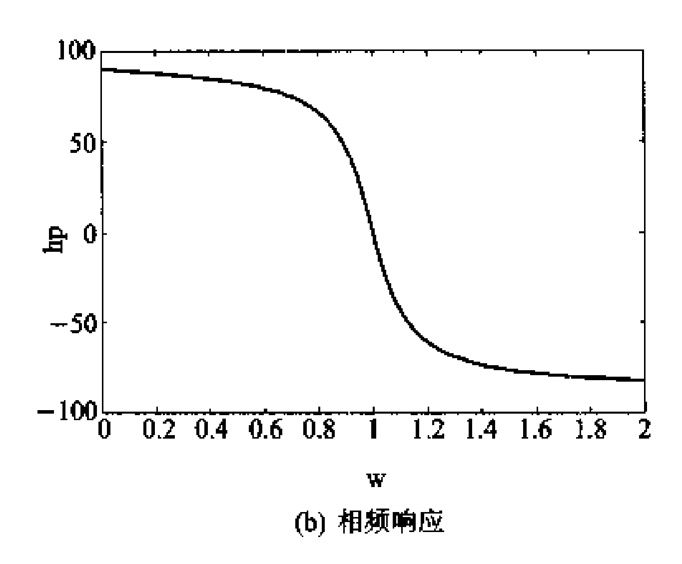
解 M 文件如下:

```
b = [1,0];
a = [1,0.2,1];
w = 0.01:0.01:2;      % w - 角频率
s = j * w;
h = polyval(b,s)./polyval(a,s);      % 计算系统函数的值
hm = abs(h);      % 计算幅值
hp = angle(h) * 180/pi;      % 相位以度为单位
h1 = figure;
plot(w,hm);xlabel('w');ylabel('hm');
h2 = figure
plot(w,hp);xlabel('w');ylabel('hp');
```

幅频响应和相频响应的曲线如图 11-3 所示。



(a) 幅频响应



(b) 相频响应

图 11－3 频率响应曲线

例 11－14 对数幅频响应曲线

已知 $H(s)=\dfrac{\frac{1}{Q}s}{s^2+\frac{1}{Q}s+1}$，若 Q 分别取 1、2、5，ω 按对数规律变换，在 0.1 rad/s至 10 rad/s 范围取值，绘制幅度分贝值的频响曲线。

解 M 文件如下：

```
w = logspace( -1,1,201);
s = j * w;
for Q = [1,2,5]
```

```
        b = [1/Q,0];
        a = [1,1/Q,1];
        h = polyval(b,s)./polyval(a,s);
        hm = 20 * log10(abs(h));
        semilogx(w,hm)
        hold on
    end
    grid
```

对数幅频响应曲线如图 11－4 所示①。

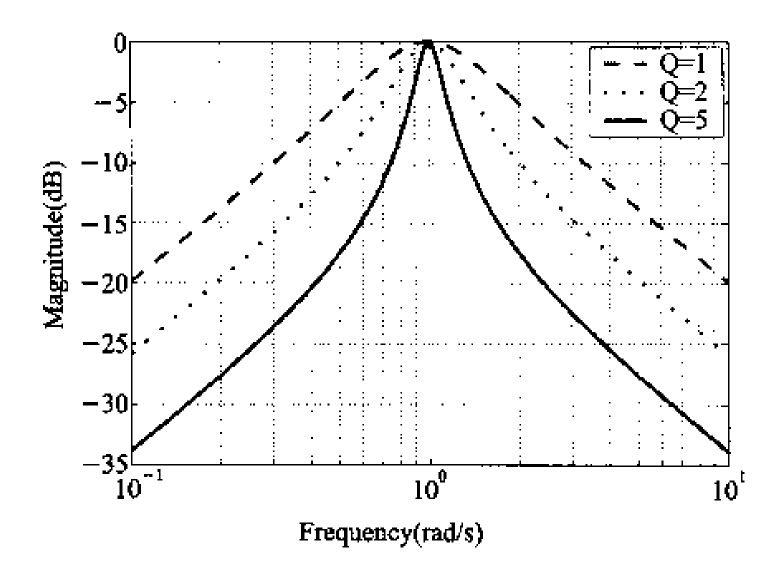


图 11－4　对数幅频响应曲线

绘制频率响应更为简便的指令为：

```
h = freqs(b,a,w)
```

其中，b 和 a 分别为系统函数分子、分母系数组成的向量；w 为角频率向量，h 为系统函数在各频率处的值。

例 11－15　用 freqs 指令绘制频率响应曲线

已知 $H(s)=\dfrac{\dfrac{1}{Q}s}{s^2+\dfrac{1}{Q}s+1}$，$Q=2$，绘制频率响应曲线。

① 由于在图形窗口中可利用下拉式菜单对图形进行编辑和添加注解，为节省纸张，在本例及后续一些程序中将省去有关指令。

解 M 文件如下：

```
w = logspace( -1,1,201);
Q =2;
b = [1/Q,0];
a = [1,1/Q,1];
freqs(b,a,w)
```

输出结果如图 11 -5 所示。

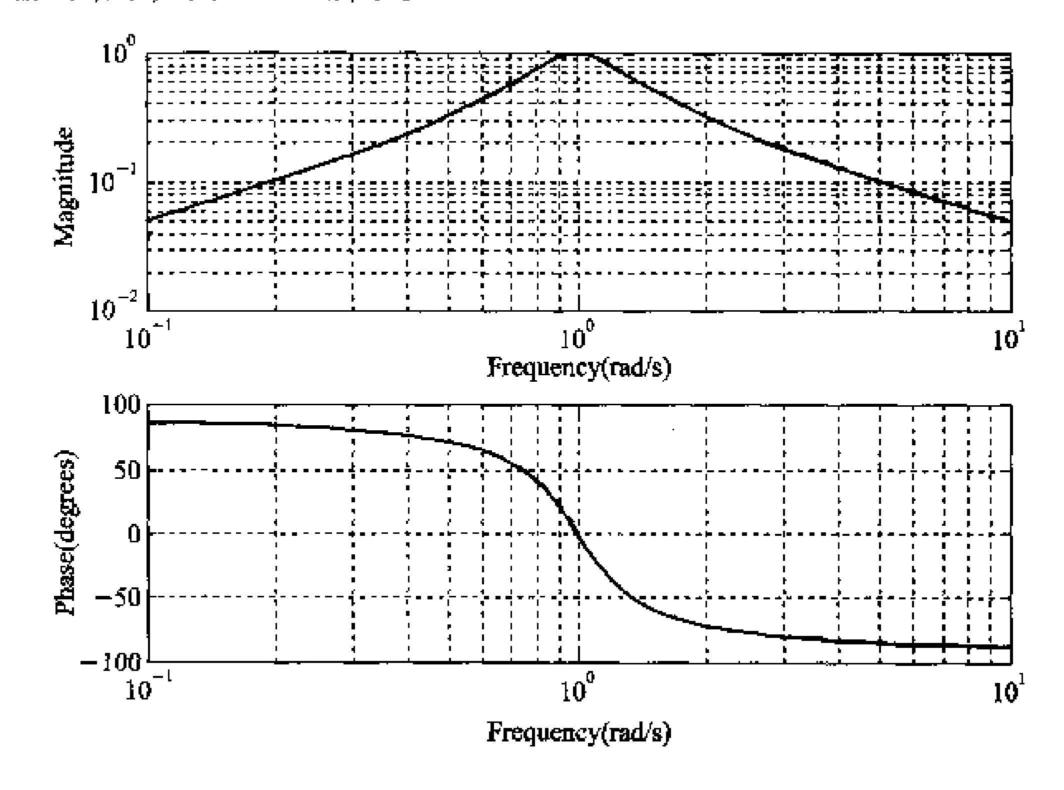


图 11 -5 使用 freqs 指令绘制的频率响应曲线

使用传递函数、极零点增益、状态空间等的系统模型绘制以分贝为单位的幅频响应和以度为单位的相频响应，可使用指令

bode(sys,w)

其中，sys 为系统模型；w 为角频率向量。如果只绘制幅频响应曲线，指令为

bodemag(sys,w)

例 11 -16 系统的幅频响应曲线

绘制系统函数 $H(s)=\dfrac{0.5s}{s^2+0.5s+1}$ 的幅频响应曲线。

解 M 文件如下：

```
w = logspace( -1,1,201);
```

```
sys = tf([0.5,0],[1,0.5,1]);
bodemag(sys,w)
grid
```

幅频响应曲线如图 11-6 所示。

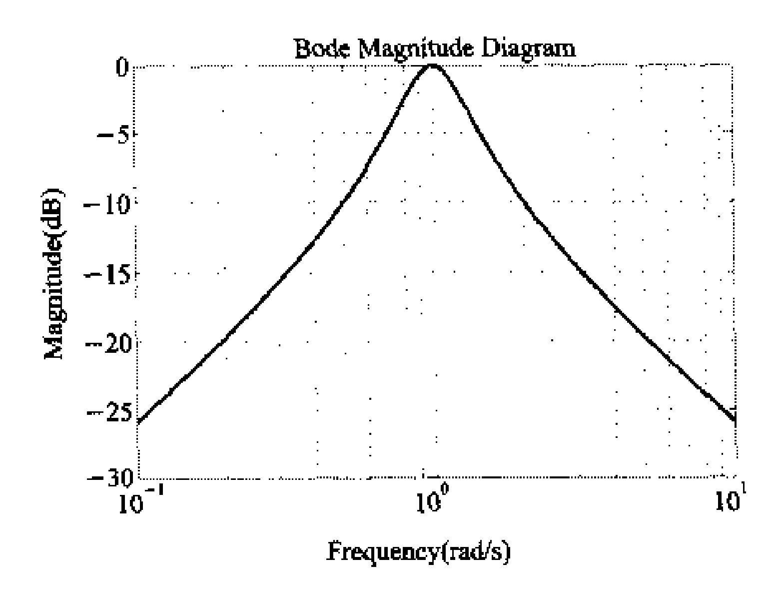


图 11-6 使用 bodemag 指令绘制的幅频响应曲线

11.5 系统的互联

系统互联的基本形式有:级联(串联)、并联和反馈连接。MATLAB 的控制系统工具箱提供了求解互联系统模型的指令。

若系统 sys1 与系统 sys2 级联,级联后系统模型的求解指令为

```
sys = series(sys1,sys2)
sys = sys1 * sys2
```

当 sys1 与 sys2 均用传递函数模型表示时,sys 也为传递函数形式。如果其中之一为状态空间模型,sys 为状态空间形式。

并联系统的系统模型求解指令为

```
sys = parallel(sys1,sys2)
sys = sys1 + sys2
```

反馈系统的求解指令为

```
sys = feedback(sys1,sys2,sign)
```

其中,sys2 为反馈子系统的模型;sign 的隐含值为 -1,表示系统为负反馈形式,

对正反馈,sign 取 1。

例 11 – 17 系统的级联

已知系统 1 与系统 2 的传递函数分别为

$$H_1(s)=\frac{3s+1}{s+2},\quad H_2(s)=\frac{1}{s}$$

求这两个系统级联后的传递函数。

解 M 文件如下:

```
sys1 = tf([3,1],[1,2]);
sys2 = tf(1,[1,0]);
sys = sys1 * sys2
```

输出结果为

```
Transfer function:
3 s + 1
--------------------------------
s^2 + 2 s
```

例 11 – 18 系统的反馈连接

求图 11 – 7 所示系统的传递函数,已知 $H_1(s)=\frac{3s+1}{s+2}$,$H_2(s)=\frac{1}{s}$。

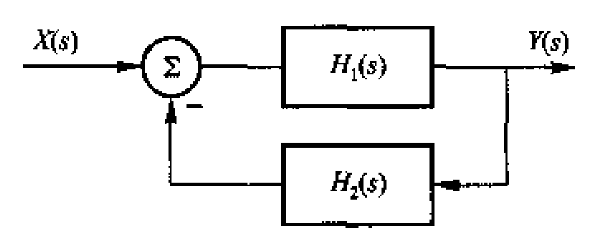


图 11 – 7 例 11 – 18 图

解 M 文件如下:

```
sys1 = tf([3,1],[1,2]);
sys2 = tf(1,[1,0]);
sys = feedback(sys1,sys2,-1)
```

输出结果为

```
Transfer function:
  3 s^2 + s
--------------------------------
s^2 + 5 s + 1
```

11.6 复频域电路分析

使用拉普拉斯变换分析电路的过程为：(1) 求解电容的初始电压和电感的初始电流；(2) 给出电路的复频域模型；(3) 建立复频域电路方程并求解；(4) 对输出量的复频域函数取逆变换。

当把电感电流也作为方程的变量时，结点方程可表示为

$$(\boldsymbol{G}+s\boldsymbol{C})\boldsymbol{X}(s)=\boldsymbol{B}(s) \tag{11-11}$$

其中，电感和电容的值在系数矩阵 $\boldsymbol{C}$ 中填入；独立电源的值及电感和电容的附加电源在 $\boldsymbol{B}(s)$ 中填入。建立式(11-11)所示方程并对其求解的过程可用函数 sana 完成，其输入变元为描述电路数据的群数组，电阻和受控电源的描述格式与第六章中的相同，独立电源的描述格式如下：

'V*** 正端 负端 s 域电压'

'I*** 正端 负端 s 域电流'

如

'V2 3 5 1/s'

表示电压源 V2 的正端连接在结点 3，负端连接在结点 5，s 域电压为 $1/s$，即时域电压为 1 V 的直流电压。s 域电压也可利用拉普拉斯变换指令 laplace 表示，上面的电压源也可写成

'V2 3 5 laplace(1,t,s)'

动态元件的描述格式为

'C*** 正端 负端 电容值 初始电压'

'L*** 正端 负端 电感值 初始电流'

'M*** 电感名1 电感名2 互感值'

其中，首字母 C 表示该元件为电容，L 表示电感，M 表示互感，电容初始电压和电感初始电流的隐含值为零。如

'C 2 3 0.1 100'

表示电容 C 的两个结点是 2 和 3，电容值为 0.1F，$t=0_-$ 时刻的电压为 100V 。

函数 sana 的完整调用格式为

[X,B,G,C] = sana(ckt)

其中，ckt 为描述电路的群数组，其他符号如式(11-11)所示。

函数 sana 为符号分析程序，元件值也可以按符号方式给出。执行函数 sana

后，程序给出结点电压、电压源电流、电感电流的 s 域解答，再利用逆拉普拉斯变换指令

```
ilaplace(xt,s,t)
```

还可得到时域解。

例 11-19 电路的 s 域分析 1

电路如图 11-8 所示，已知 $i_L(0_-)=28$ A，$v_C(0_-)=21$ V，$v_1(t)=21$ V，用拉普拉斯变换方法求电流 $i_L(t)$。

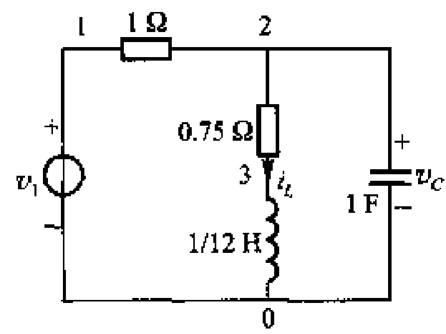


图 11-8 例 11-19 电路

解 M 文件如下：

```
ckt = {'vl 1 0 21/s'
      'rl 1 2 1'
      'r2 2 3 0.75'
      'l 3 0 1/12 28'
      'c 2 0 1 21'};
sana(ckt);
ilt = ilaplace(il);
% inverse Laplace transform of the inductor current
ilt = vpa(ilt,4)    % display the inductor current with 4 digits
```

屏幕显示结果如下：

```
Nodal voltages v** and element currents i**:
vl =21./s
v2 =(21.*s^2+182.*s+189.)/(s^3+10.*s^2+21.*s)
v3 =(-28.00-.3090e-30*s)/(s^2+10.*s+21.)
ivl =(-252.0-28.00*s)/(s^3+10.*s^2+21.*s)
il =(280.0*s+28.00*s^2+252.0)/(s^3+10.*s^2+21.*s)
ilt =12.00-12.00*exp(-7.*t)+28.00*exp(-3.*t)
```

即

$$i_L(t)=12-12e^{-7t}+28e^{-3t}\text{A}$$

例 11-20 电路的 s 域分析 2

上例中，若 $v_1(t)=\cos\left(10t+\frac{\pi}{6}\right)$V，求电流 $i_L(t)$。

解 输入电压为余弦函数，可使用 laplace 指令求其变换，但要注意，由于描述电路的数组为字符串，laplace 指令必须使用完整指令格式。即

```
'vl 1 0 laplace(cos(10*t+pi/6),t,s)'
```

然而,程序执行后会发现,求出的时域表达式不完全正确,舍入误差使解答中包含不应该有的项,如果把元件值改用分数形式表示后,解答正确。

M 文件如下:

```
ckt = {'v1 1 0 laplace(cos(10 * t + pi/6),t,s)'
       'r1 1 2 1'
       'r2 2 3 75/100'
       'l 3 0 1/12 28'
       'c 2 0 1 21'};
sana(ckt);
syms s t
ilt = ilaplace(il,s,t); % inverse Laplace transform of the inductor current
ilt = vpa(ilt,4)    % display the inductor current with 4 digits
```

计算结果为

$$i_L(t) = -20.78\,e^{-7t} + 48.79\,e^{-3t} + 0.093\,17\sin(10t) - 0.013\,62\cos(10t)\,\mathrm{A}$$

例 11-21 电路的传递函数

分析图 11-9 所示 Sallen-Key 低通滤波器的传递函数。

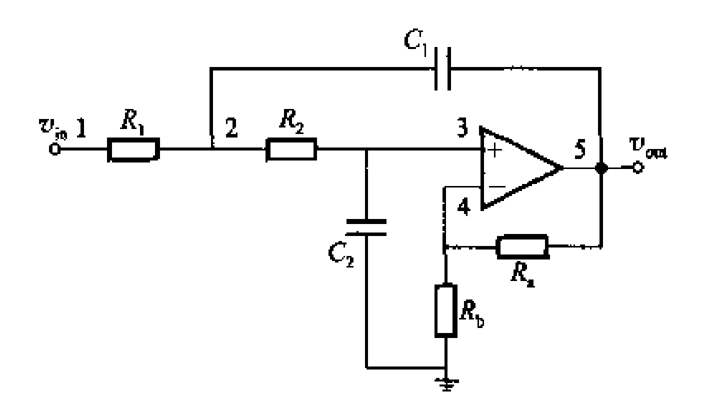


图 11-9 Sallen-Key 低通滤波器

解 由电路可看出,运算放大器和电阻 R_a、R_b 组成同相比例器,令比例系数为 k,这时,比例器用电压控制电压源描述。

M 文件如下:

```
ckt = {'vin 1 0 Vin'
       'r1 1 2 R1'
       'r2 2 3 R2'
       'c1 2 5 C1'
```

```
    'c2 3 0 C2'
    'e 5 0 3 0 k'};
x = sana(ckt);
hs = v5/v1;
pretty(hs)
```

传递函数的表达式为

```
                              k
---------------------------------------------------------------------
R2 s^2 C1 R1 C2 + (R2 C2 + C1 R1 - C1 k R1 + C2 R1) s + 1
```

当电路比较复杂时，用手工方法求解传递函数的难度较大，而计算机方法具有明显的优势。例如，运算放大器传递电压比的幅频特性并非一个恒定值，而是随频率的增加而衰减，在一定的频率范围内，运算放大器的传递函数可用一阶 s 域函数近似表示，其公式为

$$A(s)=\frac{A_0\omega_0}{s+\omega_0} \tag{11-12}$$

其中，A_0 表示直流增益；ω_0 为带宽，通常只有十几赫；$A_0\omega_0$ 称为增益带宽乘积，可用 ω_{CB} 表示，$|A(j\omega_{CB})|\approx 1$。在运算放大器手册中，增益带宽乘积为一个重要的技术参数。当电路的工作频率远高于 ω_0 时，$A(s)$ 还可近似为

$$A(s)\approx\frac{\omega_{CB}}{s} \tag{11-13}$$

在手工分析中，一般采用运算放大器的理想模型，这主要是为了简化公式推导。然而，这样做有时会掩盖电路的真实行为。

例 11－22　运算放大器的一阶模型

图 11－10 所示电路为一接地仿真电感电路，即端口特性可等效为一个电感元件，在以下两种情况下，求出端口的输入阻抗。

(1) 运算放大器采用理想模型。

(2) 当把运算放大器视为理想时，图 11－10 所示电路有多种连接形式，如每一运算放大器的两个输入端互换位置，或把 A_1 的反相端连接至结点 5，这些电路具有相同的输入阻抗。然而事实并非如此，如果采用式(11－13)所示的一阶模型对电路进行分析，当电路稳定时说明该电路在频率比较低时的确可近似为电感，而当不稳定时，该电路就不会起到仿真电感的作用，运算放大器最终会工作在饱和区，或引起电路振荡。试分析如下电路：运算放大器 A_1 的两个输入

端互换位置，即反相端接结点 1，同相端接结点 3。为了表达式简单起见，设 $R_1=R_3=R_4=R_5=R$，运算放大器的增益带宽乘积均为 ω_{GB}，试分析这种情况下电路是否能够起到仿真电感的作用。

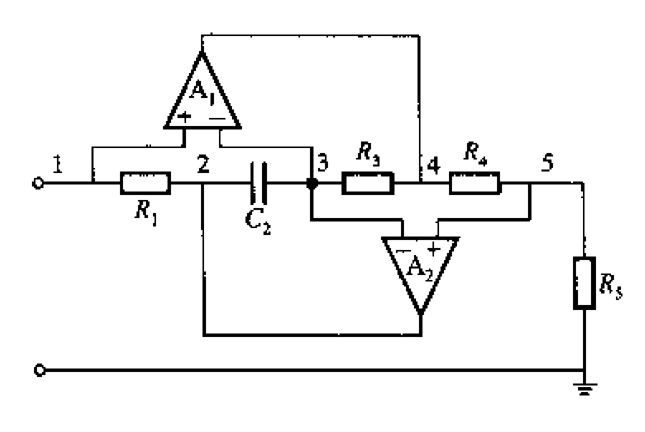


图 11-10 接地仿真电感

解 (1) M 文件如下：

```
ckt1 = {'ip 0 1 1'
        'rl 1 2 rl'
        'c2 2 3 c2'
        'r3 3 4 r3'
        'r4 4 5 r4'
        'r6 5 0 r5'
        'al 1 3 4'
        'a2 5 3 2'};
sana(cktl);
zpl = vl
```

结果显示为

```
zpl = 1/r4 * r5 * s * c2 * r3 * rl
```

由该表达式得端口的输入电感

$$L=\frac{R_1R_3R_5C_2}{R_4}$$

(2) 设在程序中用 W 表示 ω_{GB}，M 文件如下：

```
ckt2 = {'ip 0 1 1'
        'rl 1 2 r'
        'c2 2 3 c2'
```

```
        'r3 3 4 r'
        'r4 4 5 r'
        'r6 5 0 r'
        'a1 3 1 4 W/s'
        'a2 5 3 2 W/s'};
sana(ckt2);
zp2 = v1;
pretty(zp2)
```

输出结果为

```
r s (2 s^2c2 r + (2 c2 r W + 2) s - 2 W - c2 r W^2)
------------------------------------------------------------
     2 s^3 c2 r + (2 c2 r W + 2) s^2 - 2 s W - W^2
```

显然，输入阻抗分子、分母多项式的系数均有小于零的项，因而至少有一个零点和一个极点位于右半 s 平面，无论输入是电流源还是电压源，该电路都是不稳定的，故它不能起到仿真电感的作用。

例 11－23　有源 *RC* 滤波器元件参数的确定

图 11－9 所示滤波器电路，传递函数可表示为

$$H(s)=\frac{k\omega_0^2}{s^2+s\dfrac{\omega_0}{Q}+\omega_0^2}=\frac{k}{s^2\dfrac{1}{\omega_0^2}+s\dfrac{1}{Q\omega_0}+1} \tag{11-14}$$

若要求 $f_0=\dfrac{\omega_0}{2\pi}=5\ \text{kHz}$，$Q=5$，$C_1=C_2=0.01\ \mu\text{F}$，求电阻的值。

解　例 11－21 已给出，电路的传递函数为

$$H(s)=\frac{k}{s^2R_1R_2C_1C_2+s(R_2C_2+R_1C_1-kR_1C_1+R_1C_2)+1} \tag{11-15}$$

比较式(11－14)和(11－15)得

$$R_1R_2C_1C_2=\frac{1}{\omega_0^2} \tag{11-16}$$

$$R_2C_2+R_1C_1-kR_1C_1+R_1C_2=\frac{1}{Q\omega_0} \tag{11-17}$$

利用以上两式还无法求出 R_1 和 R_2，因为 k 也是未知的。这时可取一系列的 k 值进行计算，根据电阻的比值要求确定出 k 的值。当 $k=3$ 时，M 文件如下：

```
c1 = 0.01e - 6; c2 = 0.01e - 6; q = 5;
f0 = 5e3; w0 = 2 * pi * f0;
k = 3;
syms r1 r2 positive
equ1 = r1 * r2 * c1 * c2 - 1/w0^2;
equ2 = r2 * c2 + r1 * c1 - k * r1 * c1 + r1 * c2 - 1/(q * w0);
[r1,r2] = solve(equ1,equ2);
r1 = numeric(r1)
r2 = numeric(r2)
m = r2./r1
```

求得

$$R_1 = 2.881\ \text{k}\Omega, R_2 = 3.517\ \text{k}\Omega$$

电阻 R_a 和 R_b 根据 k 的值确定。因

$$k = 1 + \frac{R_a}{R_b}$$

如果取 $R_b = 10\ \text{k}\Omega$，则 $R_a = 2R_b = 20\ \text{k}\Omega$。

第十二章　z 变换

本章内容有：单边 z 变换的符号求解，逆 z 变换的部分分式展开法，差分方程的 z 变换求解，离散时间系统的传递函数模型、极零点增益模型和状态空间模型，极零点图的绘制，系统的频率响应。

12.1　z 变换

序列 $x[n]$ 的单边 z 变换定义为

$$X(z) = \sum_{n=0}^{\infty} x[n]z^{-n} \tag{12-1}$$

符号 z 变换的指令为

```
xz = ztrans(xn,n,z)
```

其中，xn 为 $x[n]$ 的符号表达式；n 为序号 n；z 为复频率 z；xz 为 $x[n]$ 的 z 变换 $X(z)$。

如果 xn 中 n 为隐含的符号变量，上面的指令也可简写成

```
xz = ztrans(xn)
```

为了避免出错，最好采用完整的指令格式。

式(12-1)也可用符号求和指令求解，格式为

```
xz = symsum(xn * z^-n,n,0,inf)
```

例 12-1　z 变换 1

求 $x_1[n] = na^n$ 和 $x_2[n] = \cos(\Omega_0 n)$ 的 z 变换。

解　用 W0 表示 Ω_0，M 文件如下：

```
syms  n  z  a  W0
x1n = n * a^n;
x2n = cos(W0 * n);
x1z = ztrans(x1n,n,z)
x2z = ztrans(x2n,n,z)
```

结果为

```
x1z = z * a/( -z + a)^2
x2z = (z - cos(W0)) * z/(z^2 - 2 * z * cos(W0) + 1)
```

例 12－2 z 变换 2

求 $x[n]=na^n\varepsilon[n-n_0]$ 的 z 变换。

解 由于 MATLAB 中没有提供阶跃序列函数，用 ztrans 指令求解本例中 $x[n]$ 的 z 变换有一定困难，但可采用 symsum 指令求解，当求和从 n_0 开始时，阶跃序列函数的值为 1，因此就不需要键入。M 文件如下：

```
syms n n0 z a
xn = n * a^n;
xz = symsum(xn * z^ - n,n,n0,inf)
```

结果为

```
xz = n0 * a^n0 * z^( - n0) * ( -1/(a - z) * z + 1/n0/(a - z)^2 * z * a)
```

逆 z 变换的符号求解指令为

```
xn = iztrans(xz,z,n)
```

其中，$x[n]$ 只适合于 $n\geqslant 0$。如果 xz 的隐含变量为 z，逆 z 变换的指令也可简写成

```
xn = iztrans(xz)
```

例 12－3 逆 z 变换

求下列函数的逆 z 变换。(1)$X_1(z)=z^{-4}$；(2)$X_2(z)=\dfrac{2z}{(z-2)^2}$。

解 M 文件如下：

```
syms n z
x1z = z^ - 4;
x2z = 2 * z/(z - 2)^2;
x1n = iztrans(x1z,z,n)
x2n = iztrans(x2z,z,n)
```

结果为

```
x1n = charfcn[4](n)
x2n = 2^n * n
```

可见，移位样值序列 $\delta[n-4]$ 在 Maple 中用 charfcn[4](n) 表示。

求逆 z 变换的程序还不完善，有时不能给出解析式。如对 $X(z)=\dfrac{z^{-N}}{z-a}$ 求逆

变换,键入以下指令

```
syms z N a
xz = z^ - N/(z - a);
xn = iztrans(xz)
```

屏幕显示为

```
xn = iztrans(z^( - N)/(z - a),z,n)
```

给出的 xn 表达式实际上不是答案,这主要是由于整数 N 在目前的符号工具箱中还无法对其说明。

再如,对 $X(z)=\dfrac{-2z^2+7z+19}{z^3+5z^2+17z+13}$求逆变换。键入以下指令

```
syms z;
b = [ -2,7,19];
a = [1,5,17,13];
xz = poly2sym(b,z)/poly2sym(a,z);
xn = iztrans(xz)
```

屏幕显示为

```
xn = 19/13 * charfcn[0](n) - ( -1)^n + 1/78 * sum(17 * (1/_alpha)^n/_alpha
     + 16 * (1/_alpha)^n,_alpha = RootOf(13 * _Z^2 + 4 * _Z + 1))
```

由于$X(z)$含有复数极点,逆变换指令未能给出其表达式。

当 $X(z)$为有理分式,而且分子、分母多项式用 z 的负幂形式表示时,有

$$X(z)=\frac{b_0+b_1z^{-1}+\cdots+b_Mz^{-M}}{a_0+a_1z^{-1}+\cdots+a_Nz^{-N}} \tag{12-2}$$

其中,$a_0\neq0$ 但 b_0、b_1 等有可能为零。利用分子系数向量 $\boldsymbol{b}$ 和分母系数向量 $\boldsymbol{a}$ 求逆变换 $x[n]$在各 n 处的值用 impz 指令,其格式为

```
[xn,n] = impz(b,a)
[xn,n] = impz(b,a,N)
[xn,t] = impz(b,a,N,fs)
impz(b,a)
```

其中,a 为分子系数向量 $\boldsymbol{a}$, $\boldsymbol{a}=[a_0,a_1,\cdots a_N]$;b 为分母系数向量 $\boldsymbol{b}$, $\boldsymbol{b}=[b_0,b_1,\cdots,b_M]$;xn 为存储$x[n]$的列向量;n 为独立变量的列向量 $\boldsymbol{n}$, $\boldsymbol{n}=[0,1,\cdots,N-1]^{\mathrm{T}}$;N 为样点的计算长度,缺省时由计算机自动确定,N 也可以是一些给定的序号向量;fs 为取样频率,即取样间隔的倒数;t 为离散时间列向量,t = n/fs。

注意,在建立式(12-2)的系数向量时,即使常数项 b_0 为零,也要在 b 中

输入。

例 12－4 根据 $X(z)$ 求序列的值

已知，$X(z)=\dfrac{z^{-1}+0.6z^{-2}}{1-1.3z^{-1}+0.4z^{-2}}$，求序列 $x[n]$ 的值，并绘制波形。设样点计算到 $n=49$。

解 M 文件如下：

```
b=[0,1,0.6];
a=[1,-1.3,0.4];
[xn,n]=impz(b,a,50)
stem(n,xn,'filled')
```

$x[n]$ 的前 10 个值为

```
xn =
        0
        1
      1.9
     2.07
    1.931
   1.6823
   1.4146
    1.166
  0.95003
  0.76861
  0.61919
```

$x[n]$ 的波形如图 12－1 所示。

当 $X(z)$ 为有理分式时，$X(z)$ 的逆变换常用部分分式法求解。如果分母多项式的 N 个根互不相等，式(12－2)可展开成

$$X(z)=\frac{r_1}{1-p_1z^{-1}}+\frac{r_2}{1-p_2z^{-1}}+\cdots+\frac{r_N}{1-p_Nz^{-1}}+k_1+k_2z^{-1}+\cdots+k_{M-N+1}z^{-(M-N)} \quad (12-3)$$

根据表 12－1，有

$$\begin{aligned}x[n]=&r_1(p_1)^n+r_2(p_2)^n+\cdots+r_N(p_N)^n+\\&k_1\delta[n]+k_2\delta[n-1]+\cdots+k_{M-N+1}\delta[n-(M-N)]\end{aligned} \quad (12-4)$$

如果分母多项式有 q 重根 p_m，相应的展开项为

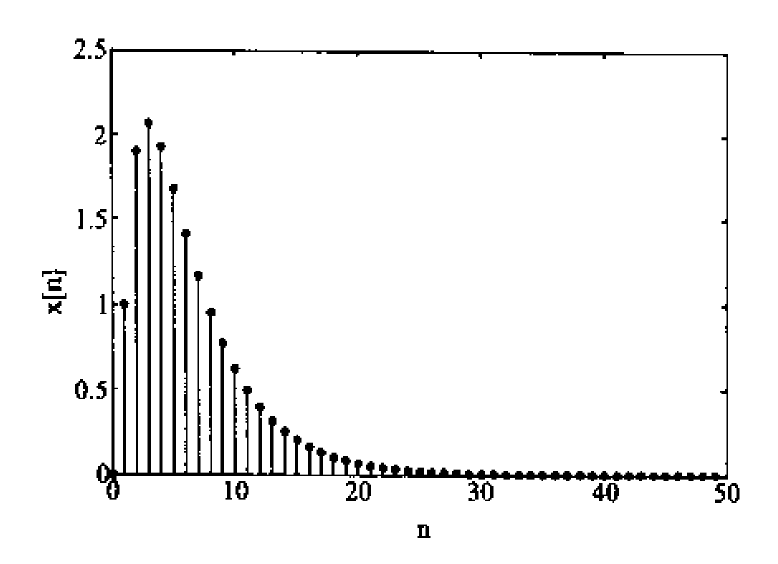


图 12-1 例 12-4 的输出波形

$$\frac{r_m}{1-p_m z^{-1}}+\frac{r_{m+1}}{(1-p_m z^{-1})^2}+\frac{r_{m+2}}{(1-p_m z^{-1})^3}+\cdots \tag{12-5}$$

上式共有 q 项。根据表 12-1，其逆变换为

$$\left[r_m+r_{m+1}(1+n)+\frac{1}{2!}r_{m+2}(1+n)(2+n)+\cdots\right](p_m)^n \tag{12-6}$$

利用指令 residuez 可求得式(12-3)和式(12-5)中的各个系数。

[r,p,k] = residuez(b,a)

其中，$\boldsymbol{b}=[b_0,b_1,\cdots,b_M]$；$\boldsymbol{a}=[a_0,a_1,\cdots,a_N]$；r 是留数列向量 $\boldsymbol{r}$，$\boldsymbol{r}=[r_1,r_2,\cdots,r_N]^T$；p 是极点列向量 $\boldsymbol{p}$，$\boldsymbol{p}=[p_1,p_2,\cdots,p_N]^T$，极点数等于 N；k 为直接项系数的行向量 $\boldsymbol{k}$，$\boldsymbol{k}=[k_1,k_2,\cdots]$。

表 12-1 基本 z 分式的逆变换

复频域	时 域
1	$\delta[n]$
$\frac{1}{z^k}$	$\delta[n-k]$
$\frac{1}{1-az^{-1}}$	a^n
$\frac{1}{(1-az^{-1})^2}$	$(1+n)a^n$
$\frac{1}{(1-az^{-1})^3}$	$\frac{1}{2}(1+n)(2+n)a^n$

续表

复频域	时 域
$\frac{1}{(1-az^{-1})^k}$	$\frac{1}{(k-1)!}\prod_{m=1}^{k-1}(m+n)a^n$
$\frac{z}{z-a}$	a^n
$\frac{z}{(z-a)^2}$	$\frac{n}{a}a^n$
$\frac{z}{(z-a)^3}$	$\frac{n(n-1)}{2!\ a^2}a^n$
$\frac{z}{(z-a)^k}$	$\frac{\prod_{m=0}^{k-2}(n-m)}{(k-1)!\ a^{(k-1)}}a^n$

例 12-5　X(z)的部分分式展开 1

用部分分式展开法求 $X(z)=\frac{80-53z^{-1}+15z^{-2}-2z^{-3}}{10-7z^{-1}+z^{-2}}$的逆变换。

解　M 文件如下：

```
b=[80,-53,15,-2];
a=[10,-7,1];
[r,p,k]=residuez(b,a)
```

执行程序的结果为

```
r =
          3
          4
p =
        0.5
        0.2
k =
     1      -2
```

根据计算结果得

$$X(z)=\frac{3}{1-0.5z^{-1}}+\frac{4}{1-0.2z^{-1}}+1-2z^{-1}$$

则逆变换

$$x[n]=3(0.5)^n+4(0.2)^n+\delta[n]-2\delta[n-1]$$

例 12-6 X(z)的部分分式展开 2

用部分分式展开法求 $X(z)=\dfrac{32z^{-1}-36z^{-2}+6z^{-3}-z^{-4}}{8-20z^{-1}+18z^{-2}-7z^{-3}+z^{-4}}$的逆变换。

解 M 文件如下：

```
b=[0,32,-36,6,-1];
a=[8,-20,18,-7,1];
[r,p,k]=residuez(b,a)
```

结果如下：

```
r =

                  1
  2.7862e-014   -9.6534e-019i
             -6   -3.311e-005i
              6   +3.311e-005i

p =

              1
            0.5
            0.5   +2.7592e-006i
            0.5   -2.7592e-006i

k =

     -1
```

其实，$X(z)$的部分展开式应该为

$$X(z)=\frac{1}{1-z^{-1}}+\frac{-6}{(1-0.5z^{-1})^2}+\frac{6}{(1-0.5z^{-1})^3}-1$$

程序求出的极点和留数不完全正确，当 $X(z)$含有高阶极点时，由于算法的缺陷造成一定的数值误差。由上式得

$$x[n]=1+\left[-6(1+n)+6\times\frac{1}{2}(1+n)(2+n)\right](0.5)^n-\delta[n]$$
$$=1+3n(1+n)(0.5)^n-\delta[n]$$

本题也可将 $X(z)$表示成如下形式

$$X(z)=z^{-1}\frac{32-36z^{-1}+6z^{-2}-z^{-3}}{8-20z^{-1}+18z^{-2}-7z^{-3}+z^{-4}}=z^{-1}X_1(z)$$

先对 $X_1(z)$进行部分分式展开，求出 $x_1[n]$，则 $x[n]=x_1[n-1]$（注意给 $x_1[n]$

乘以单位阶跃序列)。M 文件如下:

```
b=[32,-36,6,-1];
a=[8,-20,18,-7,1];
[r,p,k]=residuez(b,a)
```

输出结果为

```
r =
                1
   7.2396e-014   -1.4006e-018i
  -1.4243e-013   +1.1019e-018i
                3   +1.5585e-019i
p =
                1
              0.5
              0.5   +2.7592e-006i
              0.5   -2.7592e-006i
k =
     []
```

则 $X_1(z)$ 的展开式为

$$X_1(z)=\frac{1}{1-z^{-1}}+\frac{3}{(1-0.5z^{-1})^3}$$

于是

$$x_1[n]=\left[1+\frac{3}{2}(1+n)(2+n)(0.5)^n\right]\varepsilon[n]$$

$$x[n]=x_1[n-1]=[1+3n(1+n)(0.5)^n]\varepsilon[n-1]$$

逆 z 变换也可采用下述方法求解。将 $X(z)/z$ 的分子和分母表示为 z 的正幂形式,有

$$\frac{X(z)}{z}=\frac{\hat{b}_0z^M+\hat{b}_1z^{M-1}+\cdots+\hat{b}_M}{\hat{a}_0z^N+\hat{a}_1z^{N-1}+\cdots+\hat{a}_N} \tag{12-7}$$

在该情况下,常数项 $\hat{a}_N$ 和 $\hat{b}_M$ 一定要在系数向量中输入。对 $X(z)/z$ 展开后,给该展开式两边同乘以 z,在单极点情况下,$X(z)$ 可表示成

$$X(z)=\frac{r_1z}{z-p_1}+\frac{r_2z}{z-p_1}+\cdots+\frac{r_Nz}{z-p_N} \tag{12-8}$$

对单边 z 变换，式(12-7)中 M 一定小于 N，故 $X(z)/z$ 展开式中不会含有直接项。根据表 12-1，$X(z)$ 的逆变换为

$$x[n]=r_1(p_1)^n+r_2(p_1)^n+\cdots+r_N(p_N)^n \tag{12-9}$$

对式(12-7)进行部分分式展开的指令为

```
[r,p] = residue(bb,aa)
```

其中，bb、aa 为系数向量。

例 12-7　$X(z)$ 的部分分式

求例 12-5 中 $X(z)=\dfrac{80-53z^{-1}+15z^{-2}-2z^{-3}}{10-7z^{-1}+z^{-2}}$ 的逆变换。

解　给 $X(z)$ 的分子和分母同乘以 z^3，有

$$X(z)=\frac{80z^3-53z^2+15z-2}{z(10z^2-7z+1)}$$

$$\frac{X(z)}{z}=\frac{80z^3-53z^2+15z-2}{z^2(10z^2-7z+1)}$$

M 文件如下：

```
b = [80, -53,15, -2];
a = [10, -7,1,0,0];
[r,p] = residue(b,a)
```

输出结果为

```
r =
        3
        4
        1
       -2
p =
      0.5
      0.2
        0
        0
```

则

$$X(z)=\frac{3z}{z-0.5}+\frac{4z}{z-0.2}+1+\frac{-2}{z}$$

$$x[n]=3(0.5)^n+4(0.2)^n+\delta[n]-2\delta[n-1]$$

12.2 差分方程的z变换求解

一个离散LSI系统的差分方程可表示为

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \tag{12-10}$$

设$x[n]$为因果序列,$x[n] \overset{z}{\longleftrightarrow} X(z)$,$y[n] \overset{z}{\longleftrightarrow} Y(z)$,根据z变换的移位性质有

$$x[n-k] \overset{z}{\longleftrightarrow} z^{-k}X(z)$$

$$y[n-k] \overset{z}{\longleftrightarrow} z^{-k}Y(z) + y[-k] + y[-k+1]z^{-1} + \cdots + y[-1]z^{-(k-1)}$$

令$y_m = y[-m]$,$(m=1,2,\cdots,N)$,对式(12-10)取z变换,得

$$\sum_{k=0}^{N} a_k z^{-k} Y(z) + \sum_{k=1}^{N} \left(\sum_{m=1}^{N-k+1} a_{k+m} y_m \right) z^{-(k-1)} = \sum_{k=0}^{M} b_k z^{-k} X(z)$$

则零输入响应和零状态响应的z变换分别为

$$Y_{zi}(z) = \frac{-\sum_{k=1}^{N} \left(\sum_{m=1}^{N-k+1} a_{k+m} y_m \right) z^{-(k-1)}}{\sum_{k=0}^{N} a_k z^{-k}} \tag{12-11}$$

$$Y_{zs}(z) = \frac{\sum_{k=0}^{M} b_{k+1} z^{-k}}{\sum_{k=0}^{N} a_{k+1} z^{-k}} X(z) \tag{12-12}$$

当$X(z)$为有理分式时,利用指令 residuez 可对以上两式进行部分分式展开,从而求出时域表达式。

例12-8 差分方程的z变换求解

已知系统在$n \geq 0$时的差分方程为

$$y[n] + y[n-1] + 0.25y[n-2] = 4x[n]$$

其中,输入序列$x[n] = (0.5)^n$,初始条件$y[-1]=6$,$y[-2]=-12$。用z变换法求$y[n]$的零输入响应、零状态响应和全响应。

解 零输入响应按式(12-11)求解,零状态响应按式(12-12)求解,$x[n]$的z变换$X(z) = \dfrac{1}{1-0.5z^{-1}}$,根据$H(z)$和$X(z)$计算$Y_{zs}(z)$的分子、分母多项式使用 conv 指令,M 文件如下:

```
a=[1,1,0.25];
```

```
b = 4;
y = [6, -12]; % initial conditions
n = length(a) - 1;
% zero - input response
bzi = zeros(1,n);
for k = 1:n
    for m = 1:n - k + 1
        bzi(k) = bzi(k) - a(k + m) * y(m);
    end
end
disp('zero - input response:')
bzi
[r,p,k] = residuez(bzi,a)
% zero - state response
disp('zero - state response:')
ax = [1, -0.5];bx = 1; % X(z)
% compute Yzs(z)
azs = conv(a,ax)
bzs = conv(b,bx)
[r,p,k] = residuez(b,azs)
```

以上程序很容易修改为一个通用程序,程序执行结果如下:

```
zero - input response:
bzi =
                    -3                 -1.5
r =
        -3
         0
p =
                 -0.5
                 -0.5
k =
        []
zero - state response:
```

```
azs =
                   1          0.5          -0.25          -0.125
bzs =
     4
r =
                   1
                   2
                   1
p =
                -0.5
                -0.5
                 0.5
k =
     []
```

根据显示结果，有

$$Y_{zi}(z)=\frac{-3}{1-(-0.5)z^{-1}}$$

$$Y_{zs}(z)=\frac{1}{1-(-0.5)z^{-1}}+\frac{2}{[1-(-0.5)z^{-1}]^2}+\frac{1}{1-0.5z^{-1}}$$

于是

$$y_{zi}[n]=-3(-0.5)^n$$

$$y_{zs}[n]=[1+2(1+n)](-0.5)^n+0.5^n=(3+2n)(-0.5)^n+0.5^n$$

全响应

$$y[n]=y_{zi}[n]+y_{zs}[n]=2n(-0.5)^n+0.5^n$$

12.3 系统模型

离散时间系统的模型有：传递函数模型、极零点增益模型、状态空间模型等。传递函数模型为

$$H(z)=\frac{b_0+b_1z^{-1}+\cdots+b_Mz^{-M}}{a_0+a_1z^{-1}+\cdots+a_Nz^{-N}} \tag{12-13}$$

将传递函数中的多项式表示为 z 的正幂形式，对分子、分母作因式分解，可得系统的极零点增益模型

$$H(z)=k\frac{(z-z_1)(z-z_2)\cdots(z-z_M)}{(z-p_1)(z-p_2)\cdots(z-p_N)} \tag{12-14}$$

系统的状态空间模型为

$$\boldsymbol{x}[n+1]=\boldsymbol{A}\boldsymbol{x}[n]+\boldsymbol{B}\boldsymbol{u}[n] \tag{12-15}$$

$$\boldsymbol{y}[n]=\boldsymbol{C}\boldsymbol{x}[n]+\boldsymbol{D}\boldsymbol{u}[n] \tag{12-16}$$

其中,$\boldsymbol{x}[n]$为状态向量,$\boldsymbol{u}[n]$为输入,$\boldsymbol{y}[n]$为输出,$\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$、$\boldsymbol{D}$ 是方程的系数矩阵。对单输入单输出系统,由状态方程和输出方程可求出系统的传递函数

$$H(z)=\boldsymbol{C}(z\boldsymbol{1}-\boldsymbol{A})^{-1}\boldsymbol{B}+\boldsymbol{D} \tag{12-17}$$

其中,$\boldsymbol{1}$ 表示单位矩阵。

在 MATLAB 的控制系统工具箱中,系统可用以上任一种模型定义,其指令为

```
sys1 = filt(b,a)
sys2 = zpk(z,p,k)
sys3 = ss(A,B,C,D)
```

对以上三种模型进行转换的指令为

```
sys1 = filt(sys)
sys2 = zpk(sys)
sys3 = ss(sys)
```

其中,sys 表示系统的模型对象。

若系统 sys1 与系统 sys2 级联,级联后系统模型的求解指令为

```
sys = sys1 * sys2
```

当 sys1 与 sys2 均用传递函数模型表示时,sys 也为传递函数形式。如果其中之一为状态空间模型,sys 为状态空间形式。

求解并联系统模型的指令为

```
sys = sys1 + sys2
```

求解反馈系统模型的指令为

```
sys = feedback(sys1,sys2,sign)
```

其中,sys2 为反馈子系统的模型,sign 的隐含值为 -1,表示系统为负反馈形式,对正反馈,sign 取 1。

例 12-9 系统模型的转换 1

已知传递函数 $H(z)=\dfrac{2+1.2z^{-1}+1.46z^{-2}}{1-0.3z^{-1}-0.1z^{-2}}$,求该系统的极零点增益模型和状态空间模型。

解 M文件如下：

```
b=[2,1.2,1.46];
a=[1,-0.3,-0.1];
sys=filt(b,a)
sys2=zpk(sys)
sys3=ss(sys)
```

输出结果如下：

```
Transfer function:
2 + 1.2 z^-1 + 1.46 z^-2
 --------------------
1 - 0.3 z^-1 - 0.1 z^-2
Sampling time: unspecified
Zero/pole/gain:
2 (1 + 0.6z^-1 + 0.73z^-2)
 ---------------------
(1 - 0.5z^-1) (1 + 0.2z^-1)
Sampling time: unspecified
a =
           x1    x2
     x1   0.3   0.1
     x2     1     0
b =
           u1
     x1     2
     x2     0
c =
           x1    x2
     y1   0.9  0.83
d =
           u1
     y1     2
Sampling time: unspecified
Discrete-time model.
```

在 MATLAB 的信号处理工具箱中，对系统模型进行转换的指令见表 12－2。

表 12－2 系统模型的转换

	传递函数	极零点增益	状态空间
传递函数		[b,a] = zp2tf(z,p,k)	[b,a] = ss2tf(A,B,C,D,iu)
极零点	[z,p,k] = tf 2zp(b,a)		[z,p,k] = ss2zp(A,B,C,D,iu)
状态空间	[A,B,C,D] = tf 2ss(b,a)	[A,B,C,D] = zp2ss(z,p,k)	

表中 iu 表示第 i 个输入。

例 12－10 系统模型的转换 2

已知传递函数 $H(z)=\dfrac{2+1.2z^{-1}+1.46z^{-2}}{1-0.3z^{-1}-0.1z^{-2}}$，求 $H(z)$ 的极零点和系统的状态空间模型。

解 M 文件如下：

```
b = [2,1.2,1.46];
a = [1, -0.3, -0.1];
[z,p,k] = tf 2zp(b,a)
[A,B,C,D] = tf 2ss(b,a)
```

计算结果如下：

```
z =
          -0.3  +        0.8i
          -0.3  -        0.8i
p =
           0.5
          -0.2
k =
     2
A =
           0.3           0.1
             1             0
B =
     1
     0
```

```
C =

              1.8            1.66
D =

     2
```

绘制系统函数极零点图的指令为

```
zplane(z,p)
zplane(b,a)
```

注意:b 和 a 为传递函数分子、分母系数的行向量;而 z、p 为零点、极点的列向量。

例 12－11　极零点图

绘制 $H(z)=\dfrac{2+1.2z^{-1}+1.46z^{-2}}{1-0.3z^{-1}-0.1z^{-2}}$的极零点图。

解　M 文件如下:

```
b=[2,1.2,1.46];
a=[1,-0.3,-0.1];
zplane(b,a)
```

极零点如图 12－2 所示。

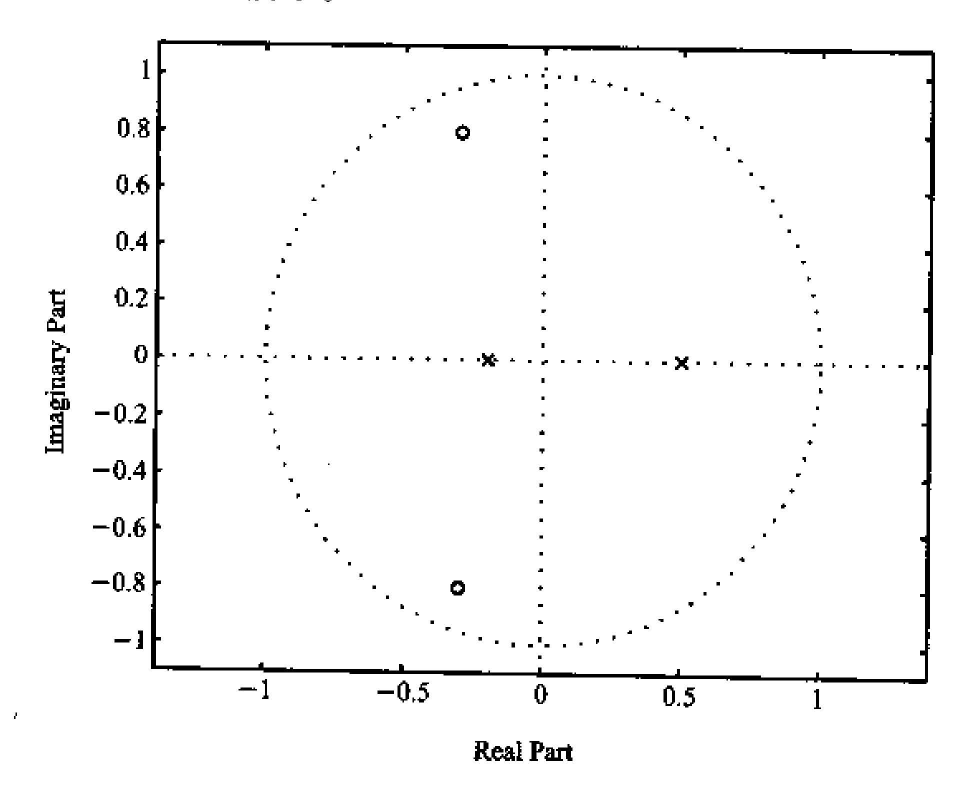


图 12－2　极零点图

12.4 频率响应

对稳定系统，若传递函数为 $H(z)$，输入序列 $x[n]=Xe^{j\Omega_0 n}(-\infty<n<\infty)$，由卷积公式可得输出序列

$$y[n]=H(z)\big|_{z=e^{j\Omega_0}}Xe^{j\Omega_0 n}=H(e^{j\Omega_0})Xe^{j\Omega_0 n} \qquad (12-18)$$

$H(e^{j\Omega_0})$一般为复数，令

$$H(e^{j\Omega_0})=|H|e^{j\angle H} \qquad (12-19)$$

当 $x[n]=|X|\cos(\Omega_0 n+\varphi_x)(-\infty<n<\infty)$时，由式(12-18)得

$$y[n]=|Y|\cos(\Omega_0 n+\varphi_y)=|H|\,|X|\cos(\Omega_0 n+\varphi_x+\angle H)$$

若用 Ω 置换 Ω_0，$H(e^{j\Omega})$的幅度 $|H|$ 随 Ω 的变化称为系统的幅频响应；$H(e^{j\Omega})$的相位$\angle H$随 Ω 的变化称为系统的相频响应。

例 12-12 频率响应曲线 1

已知传递函数

$$H(z)=\frac{2+1.2z^{-1}+1.46z^{-2}}{1-0.3z^{-1}-0.1z^{-2}}$$

绘制系统的频率响应曲线。

解 M 文件如下：

```
b=[2,1.2,1.46];
a=[1,-0.3,-0.1];
omega=linspace(-pi,pi,100);
zz=exp(-j*omega);
h=polyval(b,zz)./polyval(a,zz);
hm=abs(h);hp=angle(h)*180/pi;
subplot(2,1,1);plot(omega,hm);
xlabel('Frequency(rad)')
ylabel('Magnitude')
grid
subplot(2,1,2);plot(omega,hp);
xlabel('Frequency(rad)')
ylabel('Phase(degrees)')
grid
```



频率响应曲线如图 12 - 3 所示。

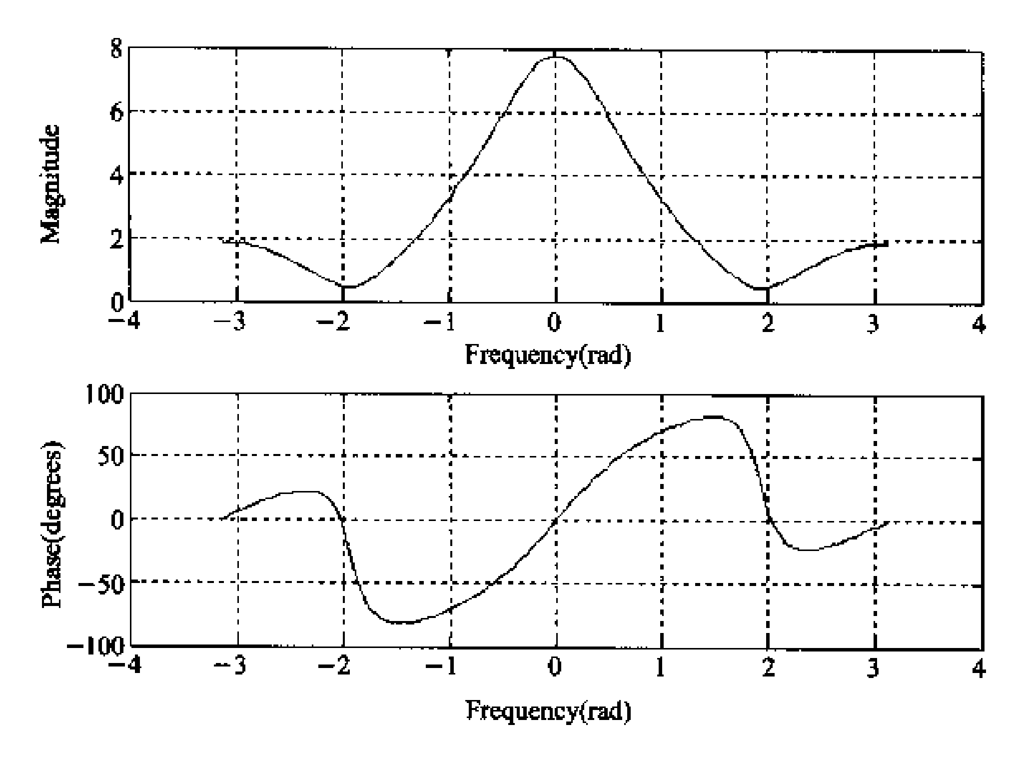


图 12 - 3　频率响应曲线

计算频率响应也可直接使用如下指令：

```
freqz(b,a)
freqz(b,a,n)
freqz(b,a,n,fs)
```

其中，b 和 a 分别为传递函数分子、分母的系数向量；n 为频率的计算点数，由于 freqz 指令采用基 2 的 FFT 算法计算频率响应，n 常取 2 的整数次幂；fs 为取样频率，当它给出时，频响曲线的横坐标为模拟频率，当它不给出时，横坐标为数字角频率，Ω 的范围为 0 到 π。

例 12 - 13　频率响应曲线 2

使用 freqz 指令绘制例 12 - 12 的频率响应曲线。

解　M 文件如下：

```
b=[2,1.2,1.46];
a=[1,-0.3,-0.1];
freqz(b,a)
```

频响曲线如图 12 - 4 所示。

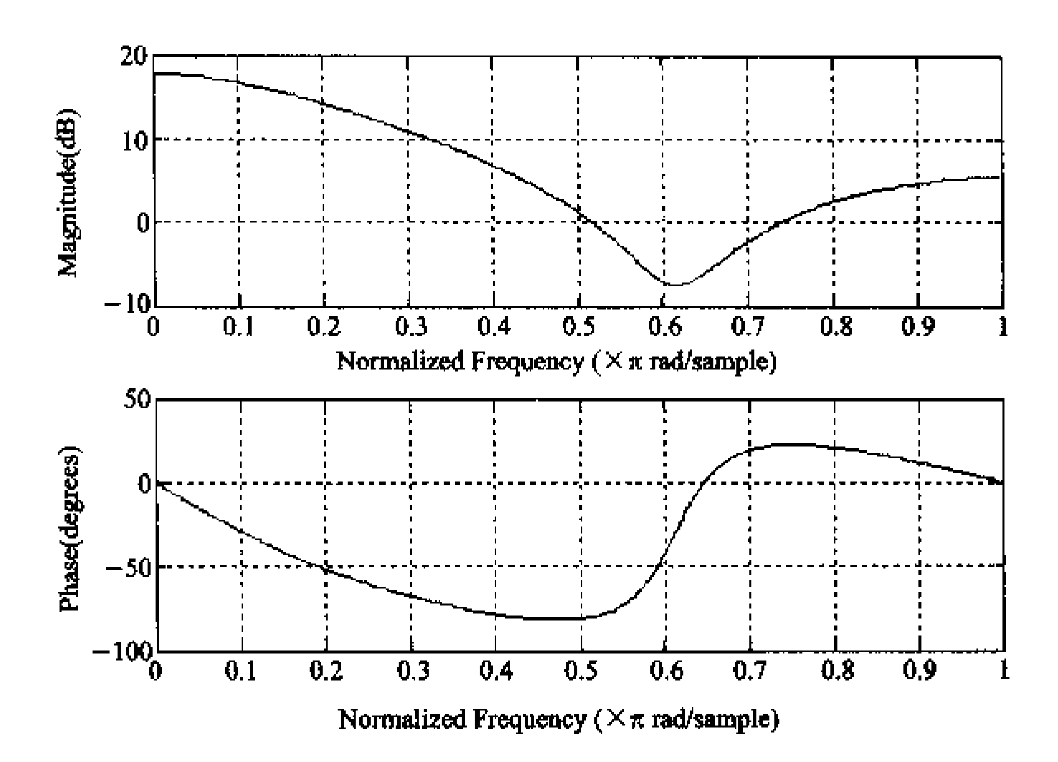


图 12-4 例 12-13 的频响曲线

第十三章　连续时间信号与系统的傅里叶分析

本章内容有：傅里叶级数和傅里叶变换的符号求解，傅里叶变换的性质，连续时间系统的频率响应，傅里叶变换与拉普拉斯变换的关系。

13.1　傅里叶级数

连续时间周期信号 $x(t)$ 可以表示成周期指数函数的和，即

$$x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\frac{2\pi}{T}t} \tag{13-1}$$

式中，T 为信号的周期；X_k 称为傅里叶系数，一般为复数，按下式计算

$$X_k = \frac{1}{T}\int_{t_0}^{t_0+T} x(t) e^{-jk\frac{2\pi}{T}t} dt \tag{13-2}$$

在用式(13-2)求解 X_k 时，往往要专门求解 X_0，它是信号 $x(t)$ 的平均值，即

$$X_0 = \frac{1}{T}\int_{t_0}^{t_0+T} x(t) dt \tag{13-3}$$

当信号 $x(t)$ 在一个周期中的函数关系给定时，利用 MATLAB 的 Symbolic Math Toolbox，根据式(13-2)和(13-3)可求出 X_k 的解析式。符号积分的指令格式为

```
int(f,t,a,b)
```

其中，f 为符号表达式；t 为积分变量；a 和 b 分别为积分的下限和上限。

例 13-1　周期矩形波的傅里叶级数

求图 13-1 所示信号的傅里叶级数，并对表达式化简。

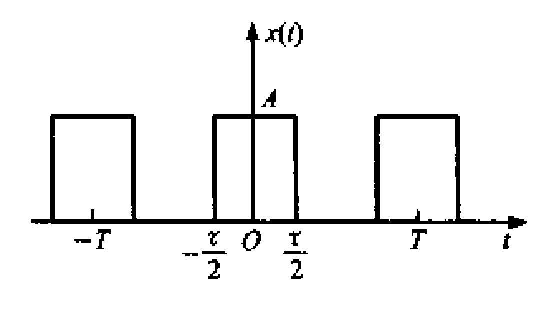


图 13-1　周期矩形波

解　求解傅里叶系数的公式如下：

$$X_0 = \frac{1}{T}\int_{-\tau/2}^{\tau/2} A\mathrm{d}t$$

$$X_k = \frac{1}{T}\int_{-\tau/2}^{\tau/2} A\mathrm{e}^{-jk\frac{2\pi}{T}t}\mathrm{d}t$$

程序中用 tao 表示τ,程序如下:

```
syms t k T tao A
x0 = int( A ,t, - tao/2 ,tao/2 )/T
f = A * exp( - j * k * 2 * pi/T * t) ;
xk = int( f ,t, - tao/2 ,tao/2 )/T;
xk = simple( xk ) ;xk
```

结果为

```
x0 = A * tao/T
xk = A * sin( tao * k * pi/T) /k/pi
```

即

$$X_0 = \frac{A\tau}{T}$$

$$X_k = \frac{A}{k\pi}\sin\left(\frac{k\pi\tau}{T}\right)$$

例 13－2　周期矩形波的谐波

设图 13－1 中 $T=8$ s,$\tau=1$ s,$A=1$,绘制该信号的频谱。

解　用符号法求出傅里叶系数 X_k 后,k 用数值向量表示,利用 subs 指令,可求得 X_k 的值。一般而言,X_k 为复数,绘制频谱图需要使用两张图。由于本例中的 X_k 为实函数,因而能在一张图上绘制频谱图。M 文件如下:

```
syms t k
T = 8; tao = 1; A = 1
x0 = int( A ,t, - tao/2 ,tao/2 )/T
f = A * exp( - j * k * 2 * pi/T * t) ;
xk = int( f ,t, - tao/2 ,tao/2 )/T;
xk = simple( xk ) ;xk
k = [ -30: -1 ,eps,1:30] ;
xk = subs( xk ,k ,'k') ;
stem( k ,xk ,'filled')
line( [ -30 30] ,[0 0] )
```

```
xlabel('k'),ylabel('Xk')
```

频谱如图 13－2 所示。

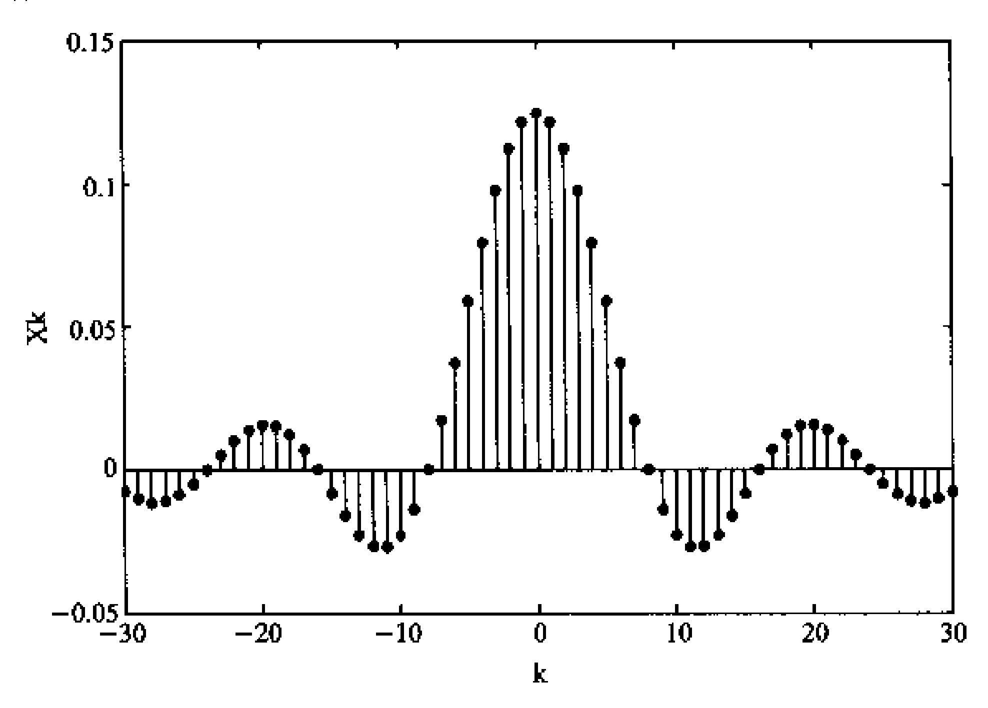


图 13－2　周期矩形波的频谱

例 13－3　周期矩形波的合成

例 13－1 给出，周期矩形波的傅里叶系数 $X_0=\frac{A\tau}{T}, X_k=\frac{A}{k\pi}\sin\left(\frac{k\pi\tau}{T}\right)$，由于 $X_k=X_{-k}$，则

$$x(t)=\sum_{k=-\infty}^{\infty}X_k e^{jk\omega_0 t}=X_0+2\sum_{k=1}^{\infty}X_k\cos(k\omega_0 t)$$

如果傅里叶级数的展开式取有限项，即

$$\hat{x}(t)=X_0+2\sum_{k=1}^{N}X_k\cos(k\omega_0 t)$$

设 $T=2$ s，$\tau=1$ s，$A=1$（方波），绘制 $N=8$ 和 $N=32$ 时 $\hat{x}(t)$ 的波形，并与 $x(t)$ 的波形进行比较。

解　写出 $\hat{x}(t)$ 和 $x(t)$ 的符号表达式，用 ezplot 函数绘制波形。程序如下：

```
syms t k
T=2;
x=sym('Heaviside(t+0.5)-Heaviside(t-0.5)');
x0=1/2;
```

```
Xk = sin(1/2 * k * pi)/k/pi;
w0 = 2 * pi/T;
xk = 2 * Xk * cos(k * w0 * t);
xcap = x0 + symsum(xk,k,1,8);
ezplot(x,[ -1,1])
hold on
ezplot(xcap,[ -1,1])
hold off
figure(2);
xcap = x0 + symsum(xk,k,1,32);
ezplot(x,[ -1,1])
hold on
ezplot(xcap,[ -1,1])
hold off
```

输出波形如图 13－3 和图 13－4 所示。

在方波的间断点时间（$|t|=0.5$ s）附近，合成波形具有比较大的上冲和下冲，相对幅度约为 0.9，这一现象称为 Gibbs 现象。

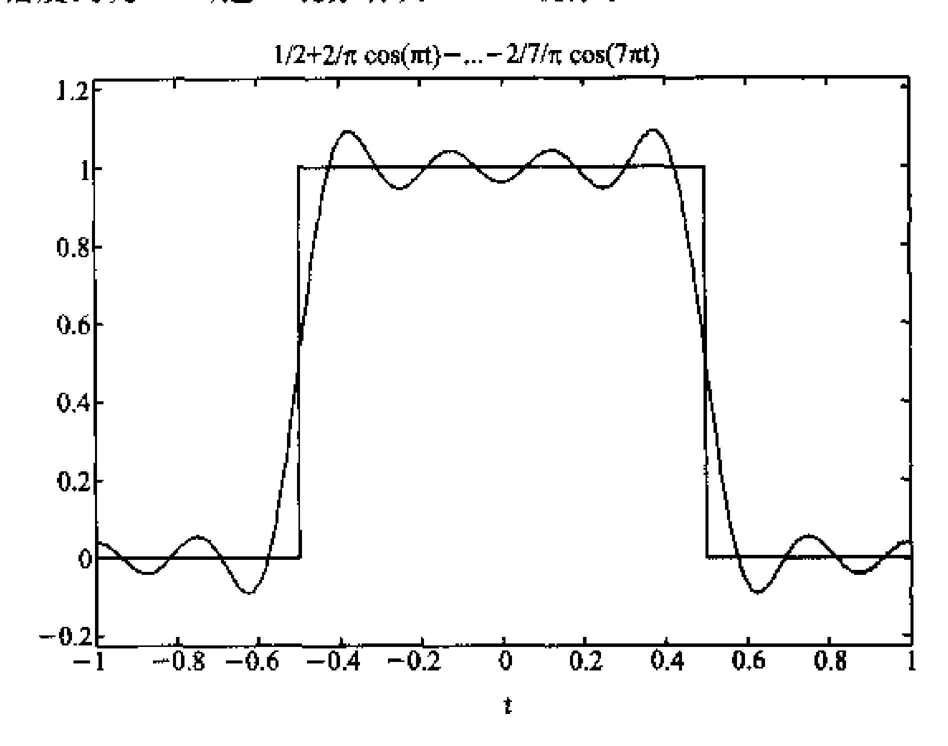


图 13－3　方波的合成（$N=8$）

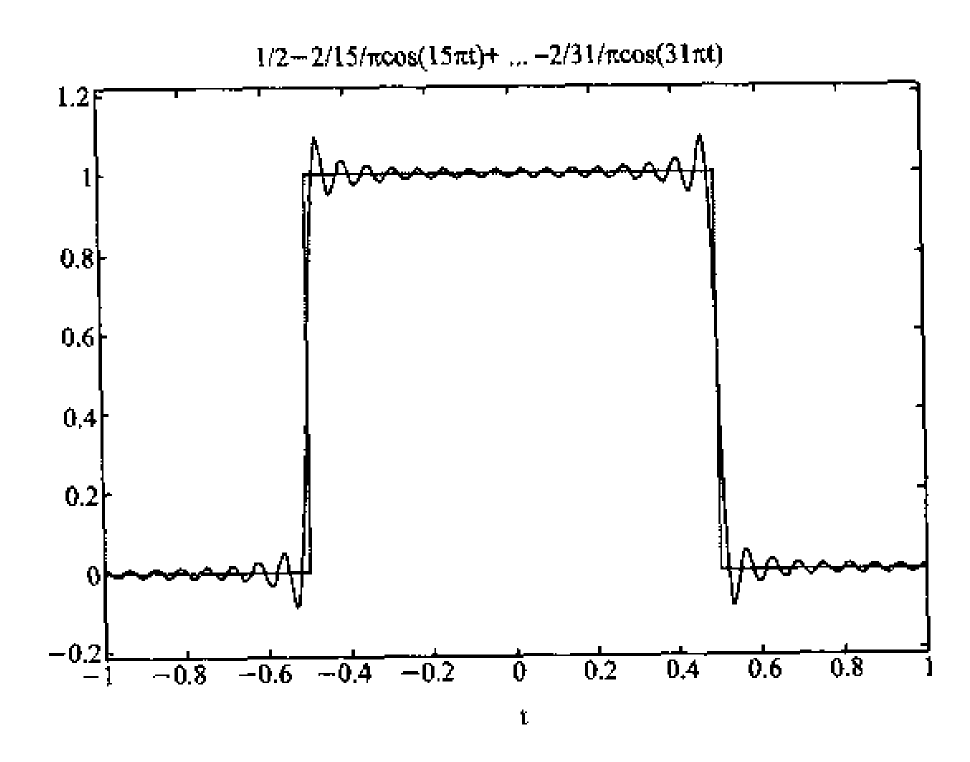


图 13－4　方波的合成($N=32$)

例 13－4　周期三角波的傅里叶级数

设周期三角波的周期 $T=1$ s，在 $|t|=T/2$ 时间范围内的函数关系为

$$x(t)=1-2|t|$$

求傅里叶级数，并对表达式化简。

解　M 文件如下：

```
syms t k T
T = 1;
x = 1 - 2 * abs(t);
x0 = int(x,t, -T/2,T/2)/T
f = x * exp( -j * k * 2 * pi/T * t);
xk = int(f,t, -T/2,T/2)/T;
xk = simple(xk);xk
```

输出结果为

```
x0 = 1/2
xk = -(cos(k * pi) -1)/k^2/pi^2
```

13.2 傅里叶变换

信号 $x(t)$ 的傅里叶变换定义为

$$X(j\omega) = \mathscr{F}[x(t)] = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt \tag{13-4}$$

逆傅里叶变换的关系式是

$$x(t) = \mathscr{F}^{-1}[X(j\omega)] = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)e^{j\omega t}d\omega \tag{13-5}$$

MATLAB 的 Symbolic Math Toolbox 提供了求解傅里叶变换的函数 fourier,格式如下:

```
X = fourier(x,t,w)
```

各标识符所表示的意义为:x 表示 $x(t)$,为符号表达式;t 表示积分变量 t;w 表示角频率 ω;X 表示 $\mathscr{F}[x(t)]$,为 $x(t)$ 的傅里叶变换。

如果符号表达式 x 中 t 为 MATLAB 规定的积分变量,而且用 w 表示复频率,上面的指令也可写成

```
X = fourier(x)
```

为了避免出错,当 xt 中除 t 外还有其他符号变量时,最好采用完整的指令格式。

逆傅里叶变换的指令格式为

```
x = ifourier(X,w,t)
x = ifourier(X)
```

如果 w 不为 MATLAB 的积分变量,求逆傅里叶变换需要采用完整的指令格式。

例 13-5 双边指数函数的傅里叶变换

求 $x(t) = e^{-3|t|}$ 的傅里叶变换。

解 由于 $x(t)$ 的表达式中不含其他符号变量,可采用傅里叶变换的简写指令格式。M 文件如下:

```
syms t
fourier(exp(-3*abs(t)))
```

结果为

```
ans = 6/(9+w^2)
```

例 13-6 矩形脉冲的傅里叶变换

设矩形脉冲的宽度用 tao 表示,求傅里叶变换。

解 M 文件如下:

```
syms t w tao
fourier(sym('Heaviside(t+tao/2)')-sym('Heaviside(t-tao/2)'),t,w)
```

结果为

```
ans = exp(1/2*i*tao*w)*(pi*Dirac(w)-i/w)-exp(-1/2*i*
tao*w)*(pi*Dirac(w)-i/w)
```

如果用 simple 指令对表达式进行化简，键入

```
simple(ans)
```

得

```
ans = 2*sin(1/2*tao*w)/w
```

例 13-7 单边指数函数的傅里叶变换

求 $x(t)=\frac{2}{3}e^{-2t}\varepsilon(t)$ 的傅里叶变换 $X(j\omega)$，并绘制 $|X(j\omega)|$ 的图形。

解 M 文件如下：

```
syms t w x;
x=2/3*exp(-2*t)*sym('Heaviside(t)');
X=fourier(x)
ezplot(abs(X));
```

输出结果为

```
X = 2/3/(2+i*w)
```

$|X(j\omega)|$ 的图形如图 13-5 所示。

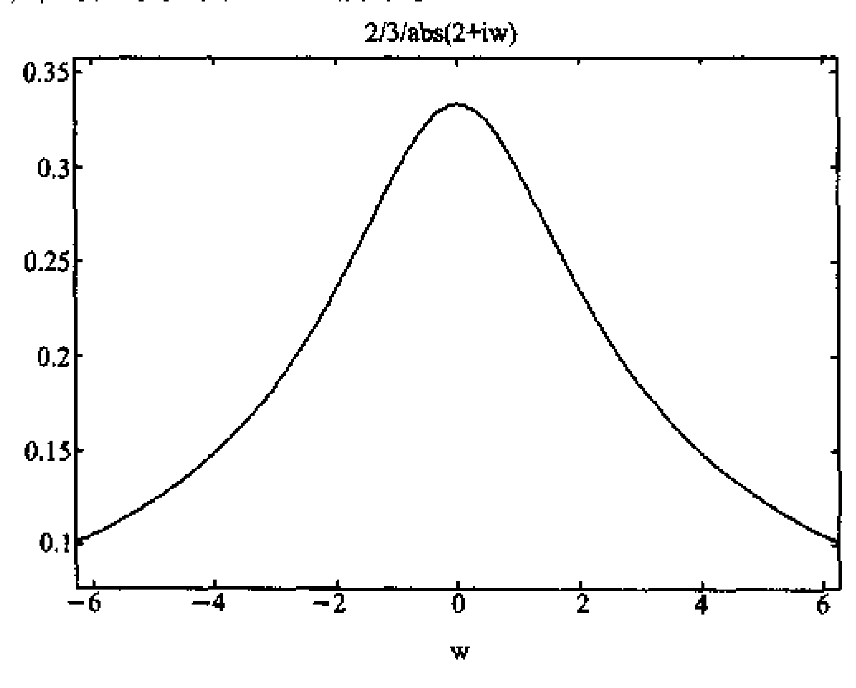


图 13-5 $x(t)=\frac{2}{3}e^{-2t}\varepsilon(t)$ 的幅度谱

例 13-8 逆傅里叶变换 1

求 $X(\mathrm{j}\omega)=\dfrac{1}{1+\omega^2}$的逆傅里叶变换。

解 M 文件如下：

```
syms w t
ifourier(1/(1+w^2),w,t)
```

输出结果为

```
ans = 1/2*exp(-t)*Heaviside(t)+1/2*exp(t)*Heaviside(-t)
```

例 13-9 逆傅里叶变换 2

求 $X(\mathrm{j}\omega)=-\mathrm{j}\dfrac{2\omega}{16+\omega^2}$的逆傅里叶变换 $x(t)$，并画出 $x(t)$的波形。

解 M 文件如下：

```
syms w,t;
X=-i*2*w/(16+w^2);
x=ifourier(X,w,t);
subplot(2,1,1);
ezplot(x);
```

$x(t)$的波形如图 13-6 所示，$x(t)$的表达式显示在图的上方。注意：由于 $x(t)$的表达式含有单位阶跃函数 Heaviside(t)，为了绘制波形，在 MATLAB 的 work 子目录中必须要有 Heaviside 函数。

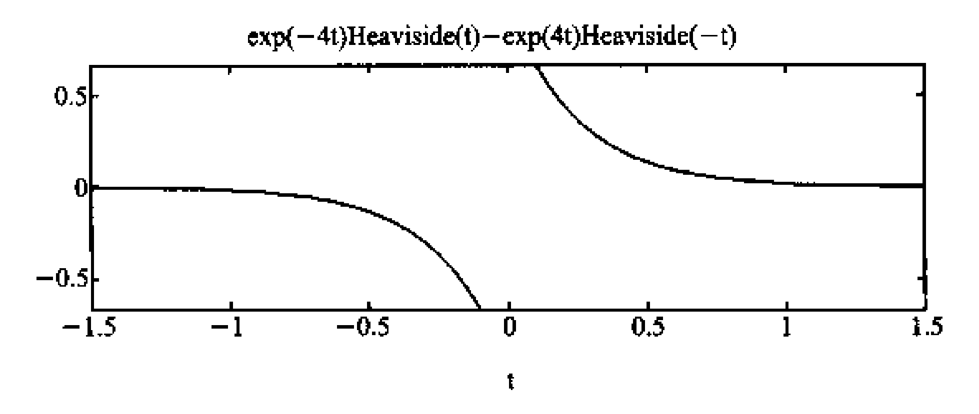


图 13-6 例 13-9 的 $x(t)$的波形

13.3 傅里叶变换的性质

1. 线性

若 $x_1(t)\longleftrightarrow X_1(\mathrm{j}\omega)$，$x_2(t)\longleftrightarrow X_2(\mathrm{j}\omega)$，则

$$a_1x_1(t)+a_2x_2(t)\longleftrightarrow a_1X_1(j\omega)+a_2X_2(j\omega) \tag{13-6}$$

式中，a_1、a_2 为任意常数。

例 13－10 傅里叶变换的线性性质

设有如图 13－7 所示的信号，它是两个矩形脉冲的叠加，即

$$x(t)=x_1(t)+x_2(t)$$

利用线性性质求 $x(t)$ 的傅里叶变换。

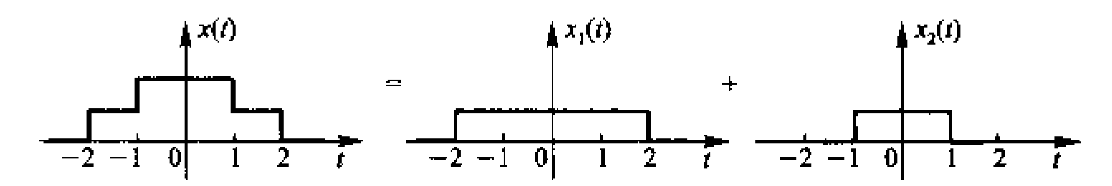


图 13－7 例 13－10 中的信号

解 M 文件如下：

```
syms t w
X1 = fourier(sym('Heaviside(t+2)') - sym('Heaviside(t-2)'),t,w);
X2 = fourier(sym('Heaviside(t+1)') - sym('Heaviside(t-1)'),t,w);
X = X1 + X2;
X1 = simple(X1)
X2 = simple(X2)
X = simple(X)
```

输出结果为

```
X1  = 2/w * sin(2 * w)
X2  = 2/w * sin(w)
X  = 2 * (sin(w) + sin(2 * w))/w
```

2. 对偶性

例 13－11 傅里叶变换的对偶性质

若 $x(t)\longleftrightarrow X(j\omega)$，用 MATLAB 求 $x_1(t)=X(t)$ 的傅里叶变换。

解 M 文件如下：

```
syms t w r
x = sym('x(t)');
X = fourier(x,t,w);
X1 = fourier(X,w,r); % the integration with respect to w
```

```
X1 = subs(X1,w,r)          % substitute r with w
```

屏幕显示为

```
X1 = 2 * pi * x( - w)
```

即

$$X(t) \longleftrightarrow 2\pi x(-j\omega)$$

3. 尺度变换

若 $x(t) \longleftrightarrow X(\omega)$，对任意不等于零的实数 a，有

$$x(at) \longleftrightarrow \frac{1}{|a|} X\left(\frac{j\omega}{a}\right)$$

例 13-12 傅里叶变换的尺度变换性质

已知信号 $x(t) = \varepsilon(t+1) - \varepsilon(t-1)$，求信号 $x(at)$ 的傅里叶变换。

解 M 文件如下：

```
syms a t w
X1 = fourier(sym('Heaviside(a * t + 1)') - sym('Heaviside(a * t - 1)'),t,w)
X1 = simple(X1)
```

结果为

```
X1 = 2 * signum(a) * sin(w/a)/w
```

由例 13-10，X = 2/w * sin(w)，则 $x(at) \longleftrightarrow \frac{1}{|a|} X\left(\frac{j\omega}{a}\right)$。

图 13-8 和图 13-9 给出了 $x(t)$ 和 $x(2t)$ 的频谱。

4. 移位

若 $x(t) \longleftrightarrow X(j\omega)$，有

$$x(t-t_0) \longleftrightarrow X(j\omega) e^{-j\omega t_0} \tag{13-7}$$

$$x(t) e^{j\omega_0 t} \longleftrightarrow X(j\omega - j\omega_0) \tag{13-8}$$

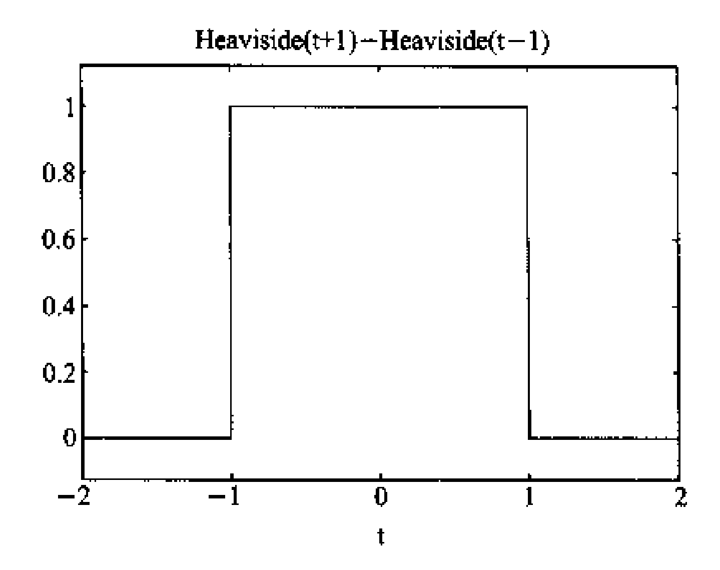
式(13-7)和式(13-8)分别称为傅里叶变换的时移性质和频移性质。

时移性质表明，信号在时间轴上的移位，其频谱函数的幅度谱不变，而相位谱产生附加相移 ωt_0。

频移性质表明，若要使一个信号的频谱在频率轴上右移 ω_0 单位，在时域就对应于其时间信号 $x(t)$ 乘以 $e^{j\omega_0 t}$。

例 13-13 傅里叶变换的移位性质

设有信号 $x(t) = \frac{1}{2} e^{-3t} \varepsilon(t)$，$x_1(t) = x(t-0.2) = \frac{1}{2} e^{-3(t-0.2)} \varepsilon(t-0.2)$，试用 MATLAB 绘出 $x(t)$ 和 $x_1(t)$ 的波形及频谱。



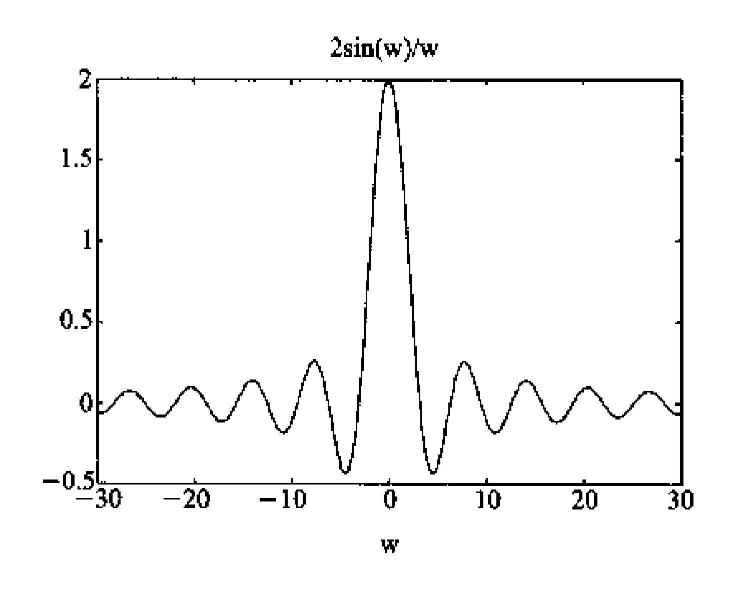
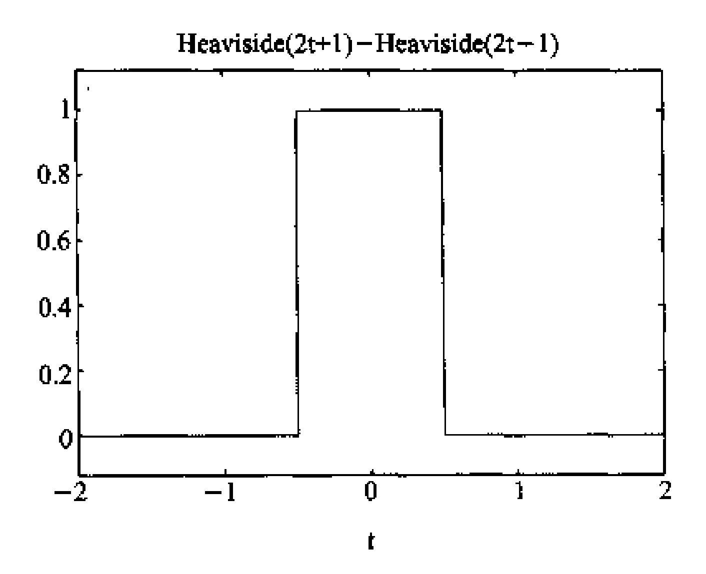


图 13-8 $x(t)$及其频谱

解 M 文件如下：

```
syms t w
x = 1/2 * exp( -3 * t) * sym('Heaviside(t)');
X = fourier(x,t,w);
r = [ -8:0.02:8];
X = subs(X,r,w);
subplot(3,1,1); ezplot(x,[ -2,2]);grid;
subplot(3,1,2); plot(r,abs(X));
```



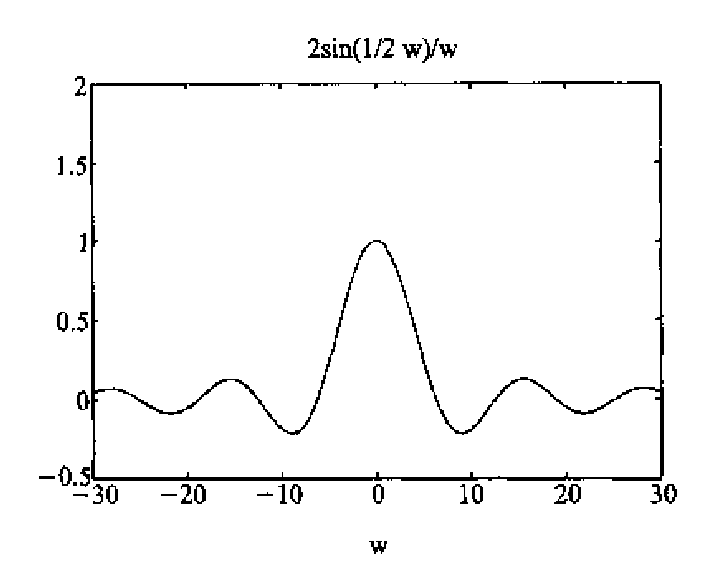


图 13－9 $x(2t)$及其频谱

```
xlabel('w'),ylabel('Magnitude'),grid;
P = angle(X) * 180/pi;
subplot(3,1,3); plot(r,P);
xlabel('w'),ylabel('Phase'),grid;
% ************************
x = subs(x,t-0.2,t);
figure(2)
X = fourier(x,t,w);
```

```
r = [ -8:0.02:8];
X = subs(X,r,w);
subplot(3,1,1); ezplot(x,[ -2,2]);grid;
subplot(3,1,2); plot(r,abs(X));
xlabel('w'),ylabel('Magnitude'),grid;
P = angle(X) * 180/pi;
subplot(3,1,3); plot(r,P);
xlabel('w'),ylabel('Phase'),grid;
```

$x(t)$和$x_1(t)$的波形及频谱如图13-10和13-11所示。可见,$x_1(t)$的幅度谱与$x(t)$的相同,而相位谱附加的相移度数为$\omega t_0 \times 180°/\pi \approx 11.46\omega$。

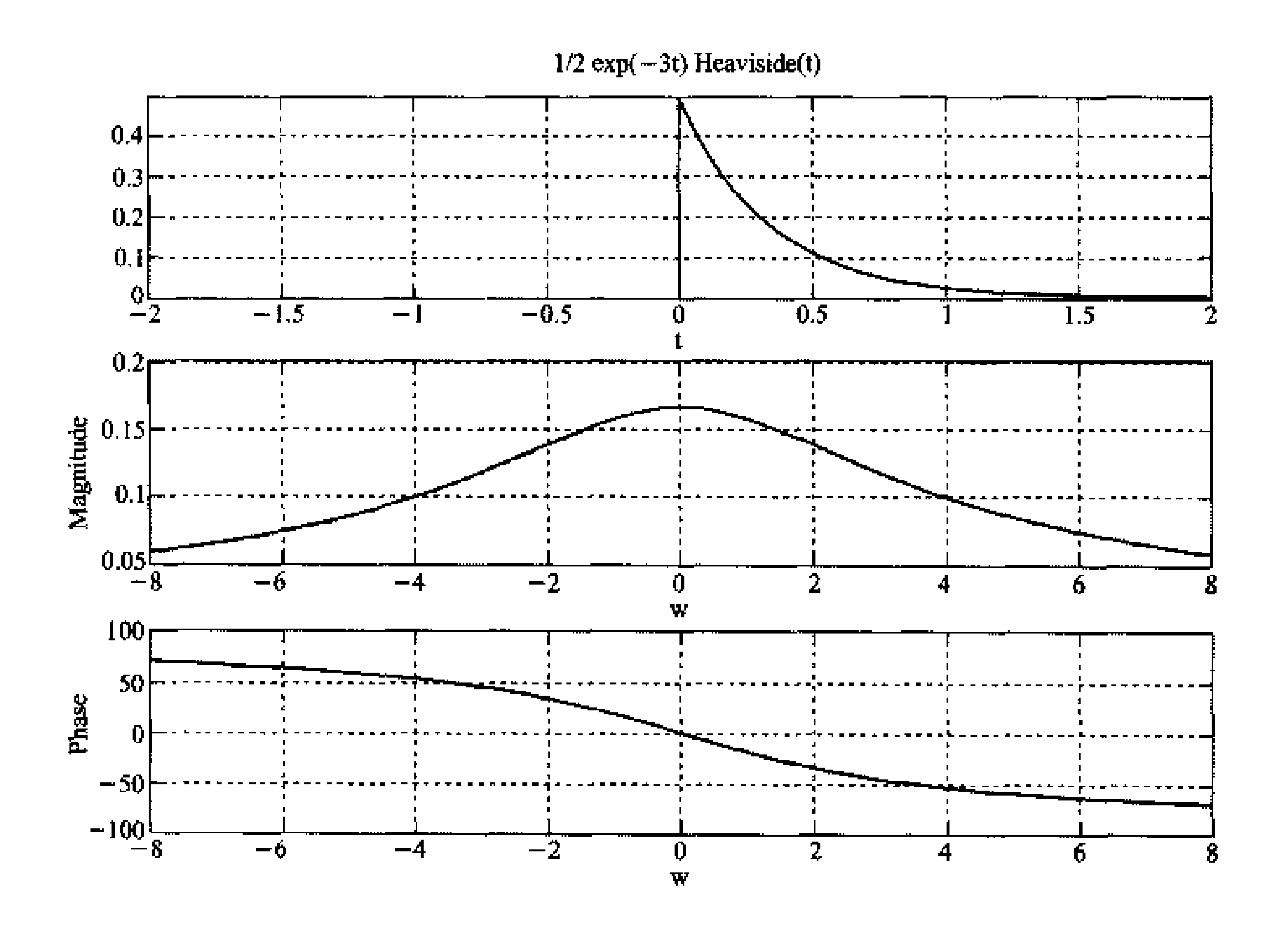


图13-10　$x(t)$的波形及频谱

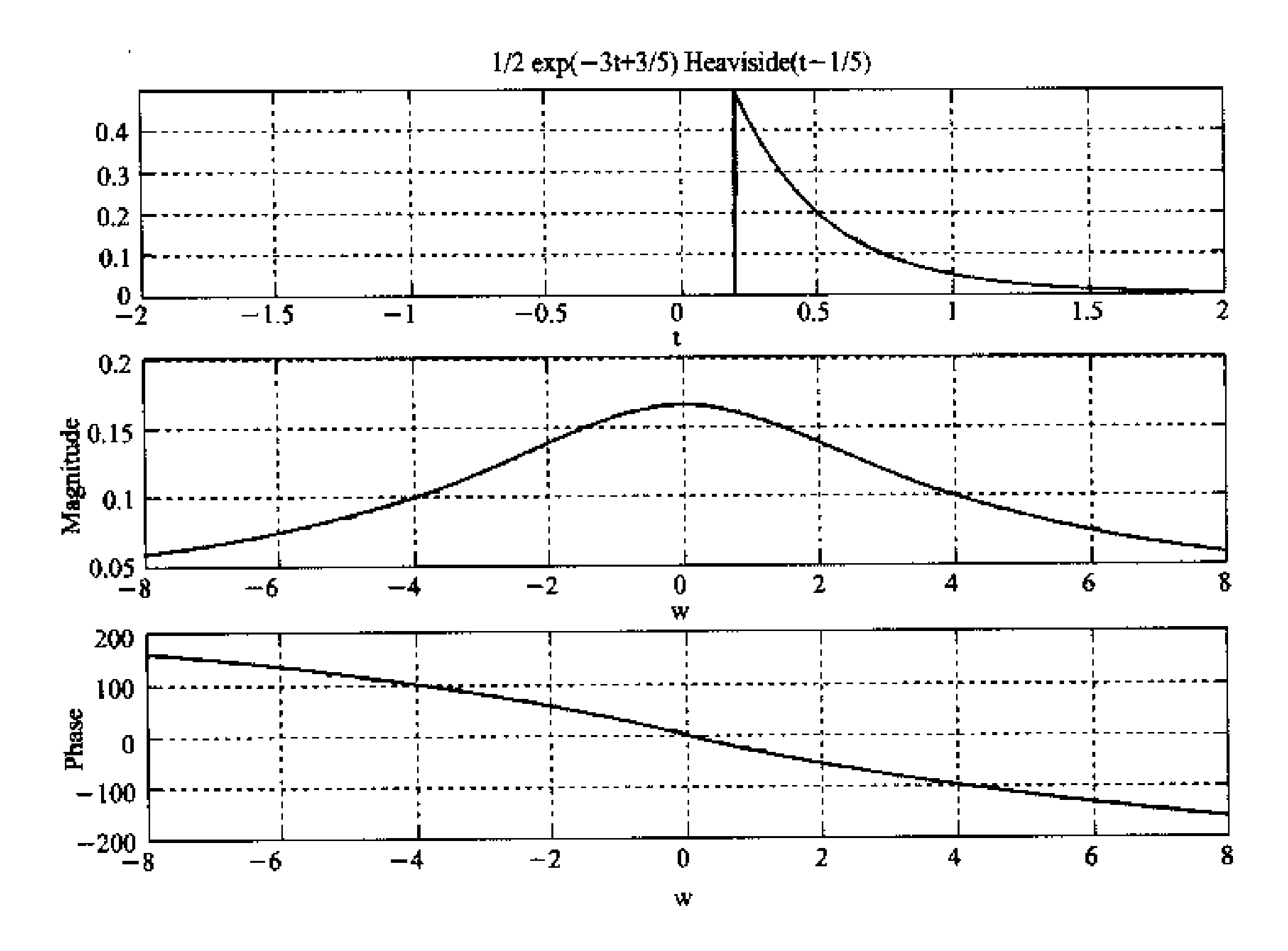


图 13-11　$x_1(t)$的波形及频谱

13.4　连续时间系统的频域分析

频域分析是把系统的激励与响应之间的关系应用傅里叶变换从时域中变换到频域中进行考察的一种方法。这种方法与相量法、拉普拉斯变换法的思想方法相似，把在时域中处理时间变量 t 转换成处理频率变量 ω，从解系统的线性微分方程转化为解代数方程，并通过频域系统函数来研究响应的频率组成和系统的功能。

利用频域分析法求解系统的零状态响应通常会遇到两类问题。一类是已知系统的冲激响应或系统函数，求任意非周期信号激励下的响应；另一类是已知系统的结构、元件约束求响应。

对于前一类问题，若输入为 $x(t)$，冲激响应为 $h(t)$，则响应

$$y(t)=h(t)*x(t)$$

在频域分析法中,若 $x(t)\longleftrightarrow X(\mathrm{j}\omega)$、$h(t)\longleftrightarrow H(\mathrm{j}\omega)$ 及 $y(t)\longleftrightarrow Y(\mathrm{j}\omega)$,利用傅里叶变换的时域卷积定理,有

$$Y(\mathrm{j}\omega)=H(\mathrm{j}\omega)X(\mathrm{j}\omega) \tag{13-9}$$

$H(\mathrm{j}\omega)$称为频域系统函数,与 s 域系统函数 $H(s)$不同的是其以傅里叶变换的形式给出。

若从物理概念的角度来理解频域分析法,就是输入信号的频谱函数,经过系统的处理后发生了改变,由原来的 $X(\mathrm{j}\omega)$变成了 $Y(\mathrm{j}\omega)=H(\mathrm{j}\omega)X(\mathrm{j}\omega)$。系统的作用相当于对输入信号的各频率分量进行加权,某些频率的分量有可能增强,而另一些分量则相对削弱或保持不变。而且,经过系统处理的每一个频率分量都会产生各自的相移。每一个频率分量改变的规律完全由系统函数 $H(\mathrm{j}\omega)$所决定。因此,在输入信号频谱给定的情况下,要想得到需要的输出频谱的过程,实际上就是对 $H(\mathrm{j}\omega)$进行设计的过程。所以,频域分析法并不仅仅是一种计算响应的方法,更深层次上的意义是该方法能够使设计者在频域中通过输入、输出信号的频谱清晰地看到系统对信号每一个分量的变换过程及对 $H(\mathrm{j}\omega)$的要求,从而获得系统函数的信息。

一般情况下,频域分析法中的系统函数可表示为

$$H(\mathrm{j}\omega)=\frac{b_0(\mathrm{j}\omega)^M+b_1(\mathrm{j}\omega)^{M-1}+\cdots+b_{M-1}(\mathrm{j}\omega)+b_M}{a_0(\mathrm{j}\omega)^N+a_1(\mathrm{j}\omega)^{N-1}+\cdots+a_{N-1}(\mathrm{j}\omega)+a_N} \tag{13-10}$$

式中的系数 $a_0,a_1,\cdots,a_N$ 和 $b_0,b_1,\cdots,b_M$ 取决于系统的结构,而与输入无关。可见,$H(\mathrm{j}\omega)$是 ω 的复函数,可写作

$$H(\mathrm{j}\omega)=|H(\mathrm{j}\omega)|\,\mathrm{e}^{\mathrm{j}\varphi(\omega)} \tag{13-11}$$

式中,$|H(\mathrm{j}\omega)|$称为系统函数的幅度函数;$\varphi(\omega)$称为相位函数。

应用频域分析法分析稳定系统,意味着将输入信号分解为无穷多项 $\mathrm{e}^{\mathrm{j}\omega t}$信号的叠加,分别计算系统对每一 $\mathrm{e}^{\mathrm{j}\omega t}$分量的响应再求和即可得出完整的响应信号。这一观点在线性时不变系统的分析方法中已得到了反复运用。所以,在应用频域分析法求解具体电路时,电路元件的频域模型与相量法的类似,其区别是相量法中通常是某一频率分量,而在频域法中的 ω 是频率轴上的连续变量(从 $-\infty$ 到∞)。

MATLAB 提供了计算连续系统频率响应 $H(\mathrm{j}\omega)$的函数 freqs,该函数可以求出系统频率响应的数值解,并可绘出系统的幅频和相频响应曲线。freqs 函数的调用格式如下:

```
h = freqs(b,a,w)
```

其中，b 为对应于式(13－10)中的向量$[b_0,b_1,b_2,\cdots,b_M]$；a 为向量$[a_0,a_1,a_2,\cdots,a_N]$；w 为角频率向量。向量 h 则返回在向量 w 所定义的频率点上系统函数的值。

[h,w] = freqs(b,a)

计算默认频率范围内 200 个频率点的系统函数的值，用 h 表示，200 个角频率点记录在 w 中。

[h,w] = freqs(b,a,n)

计算默认频率范围内 n 个频率点上系统函数的值。

freqs(b,a)

该格式并不返回系统函数的值，而是以对数坐标的方式绘出系统频率响应的曲线。

例 13－14 电路的频率响应

图 13－12 是一无源二阶低通滤波器，系统函数 $H(j\omega)$为

$$H(j\omega)=\frac{V_2(j\omega)}{V_1(j\omega)}=\frac{1}{1-\omega^2LC+j\omega\dfrac{L}{R}}$$

设 $L=0.4$ H，$C=0.002$ F，$R=\sqrt{\dfrac{L}{2C}}=10\Omega$，试绘制频率响应曲线。

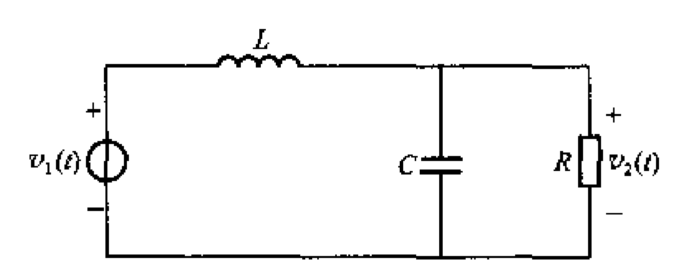


图 13－12 二阶低通滤波器

解 令通带频率 $\omega_c=\dfrac{1}{\sqrt{LC}}=35.4$ rad/s，当 $\omega=\omega_c$ 时，

$$|H(j\omega_c)|=\frac{1}{\sqrt{2}}|H(0)|=\frac{1}{\sqrt{2}}$$

将元件值代入 $H(j\omega)$的表达式，得

$$H(j\omega)=\frac{1}{0.0008(j\omega)^2+0.04(j\omega)+1}$$

绘制频率响应曲线的程序如下：

```
b = 1;
a = [0.0008,0.04,1];
freqs(b,a);
```

程序运行结果如图 13－13 所示。

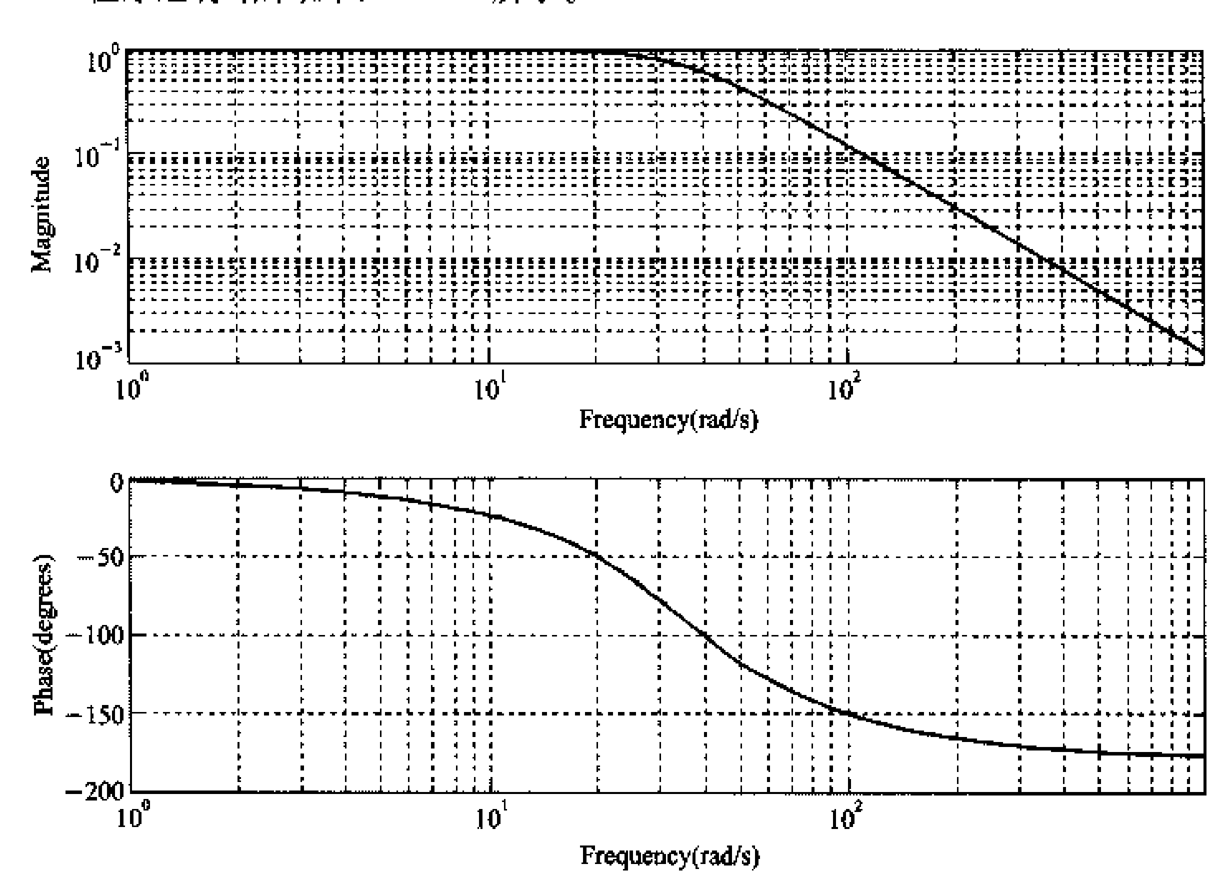


图 13－13　二阶低通滤波器的频率响应

例 13－15　系统函数的频率响应

已知 $H(j\omega)=\dfrac{14-j\omega}{14+j\omega}$,求频率响应。

解　M 文件如下:

```
b = [-1,14];
a = [1,14];
[h,w] = freqs(b,a,100);
mag = abs(h);
phase = angle(h) * 180/pi;
subplot(211);
```

```
plot(w,mag);
grid
xlabel('w');
ylabel('Magnitude');
subplot(212);
plot(w,phase);
grid
xlabel('w');
ylabel('Phase(degrees)');
```

程序运行结果如图 13－14 所示，在任一频率处系统函数的幅度均为 1。

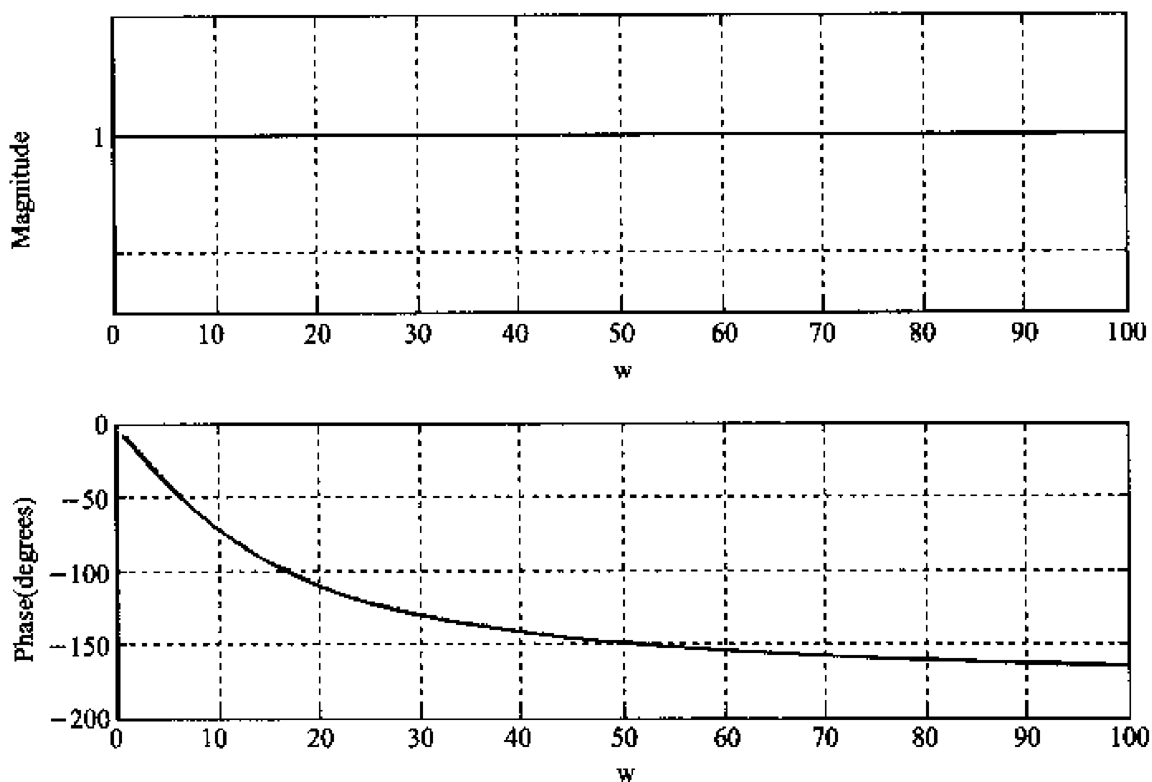


图 13－14　例 13－15 的频率响应

13.5　傅里叶变换与拉普拉斯变换的关系

对因果信号 $x(t)$，傅里叶变换和拉普拉斯变换的公式分别为

$$X_{FT}(j\omega) = \int_{0-}^{\infty} x(t)e^{-j\omega t}dt \tag{13-12}$$

$$X_{LT}(s) = \int_{0-}^{\infty} x(t)e^{-st}dt \tag{13-13}$$

当 $x(t)$ 满足绝对可积条件时，傅里叶变换 $X_{FT}(j\omega) = X_{LT}(s)\big|_{s=j\omega}$。当信号 $x(t)$ 不满足绝对可积条件时，若拉普拉斯变换的收敛域不包括 $j\omega$ 轴，该信号根本就不存在傅里叶变换，所以不能由拉普拉斯变换寻求傅里叶变换。

例 13-16　拉普拉斯变换的曲面图和傅里叶变换的频谱

设有信号 $x(t) = \varepsilon(t) - \varepsilon(t-2)$，试用 MATLAB 绘制该信号拉普拉斯变换的曲面图和傅里叶变换的频谱。

解　容易求得该信号的拉普拉斯变换和傅里叶变换分别为

$$X_{LT}(s) = \frac{1-e^{-2s}}{s}, \quad X_{FT}(j\omega) = 2\mathrm{sinc}(\omega)e^{-j\omega}$$

M 文件如下：

```
% 绘制拉普拉斯变换曲面图
clf;
a = -0:0.1:5;
b = -20:0.1:20;
[a,b] = meshgrid(a,b);
s = a + i * b;
xs = (1 - exp( -2 * s)) ./s;
xs = abs(xs);
mesh(a,b,xs);
surf(a,b,xs);
view( -60,20);
axis([ -0,5, -20,20,0,2]);
title('信号的拉普拉斯变换');
colormap(hsv);
% 绘制傅里叶变换频谱图
figure(2)
w = -20:0.1:20;
xw = 2 * sinc(w/pi). * exp( -i * w);
plot(w,abs(xw));
```

title('信号的傅里叶变换');

输出结果如图 13-15 和图 13-16 所示。可见,拉普拉斯变换的曲面图在截面 Re(s)=0 上的曲线为傅里叶变换的频谱。

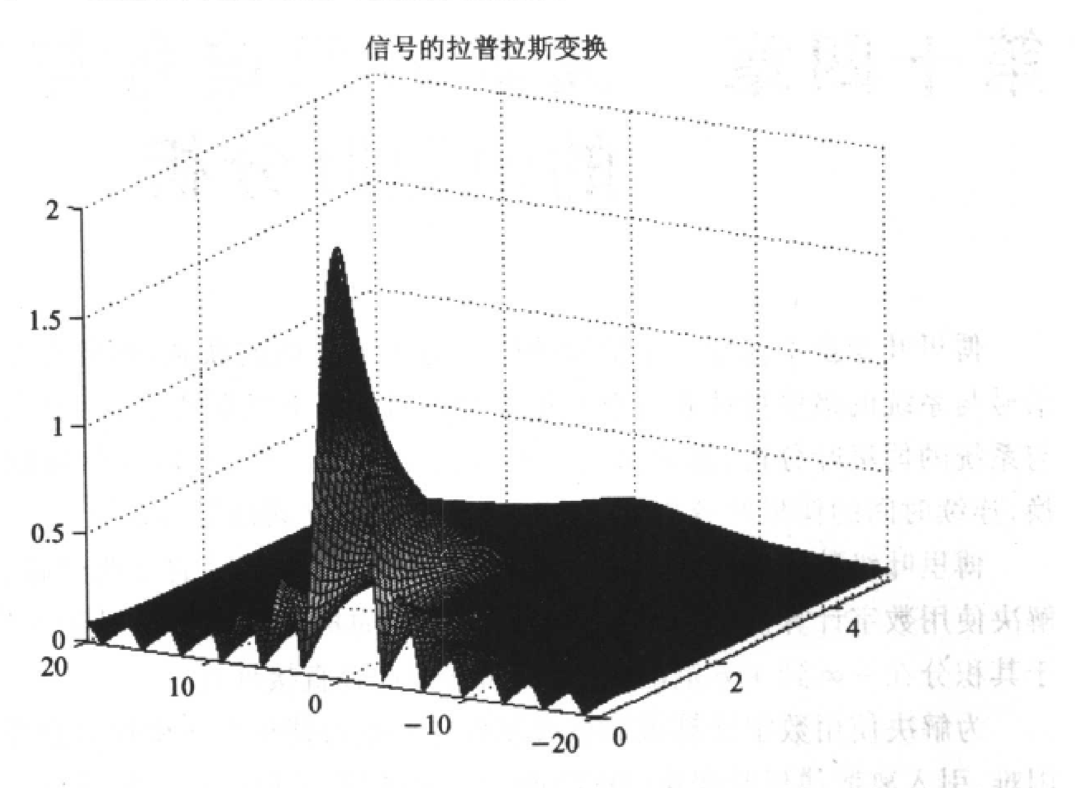


图 13-15 信号拉普拉斯变换的曲面图

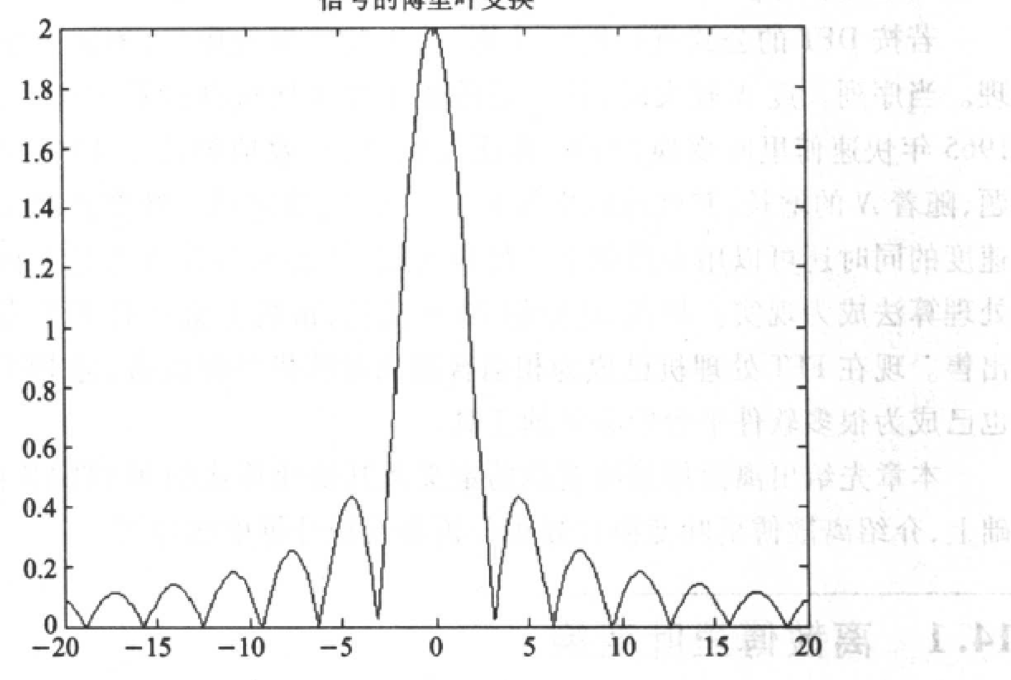


图 13-16 信号傅里叶变换的频谱

第十四章　离散时间信号与系统的傅里叶分析

傅里叶变换不仅建立了信号的时域与频域之间的联系，而且对分析和研究信号与系统的诸多特性起着至关重要的作用。第十三章已介绍了连续时间信号与系统的傅里叶分析，如果对连续时间信号进行取样，对取样信号取傅里叶变换，连续时间的傅里叶分析方法也可推广至离散时间信号与系统。

傅里叶级数或变换从理论上解决了连续与离散信号的分析问题，但还没有解决使用数字计算机的计算方法问题。以非周期连续时间信号的变换为例，由于其积分在 $-\infty$ 到 $+\infty$ 范围积分，因而它不适合直接计算。

为解决使用数字计算机编程或硬件电路进行傅里叶正变换或逆变换遇到的困难，引入离散傅里叶变换（DFT）的定义和性质，是使用计算机进行傅里叶分析必要的理论依据。

若按 DFT 的公式直接进行计算，由于其计算速度慢，不适合信号的实时处理。当序列长度 N 较大时，计算量的快速增加使这种计算方法失去实际价值。1965 年快速傅里叶变换（FFT）算法的发现，有效地解决了 DFT 的实时计算问题，随着 N 的增长，其优越性更加显著。另外，快速傅里叶变换算法在提高计算速度的同时还可以用专用数字硬件来实现，使得原来曾认为是不切实际的信号处理算法成为现实。早在 20 世纪 70 年代初，市场上就已有 FFT 专用硬件电路出售。现在 FFT 处理机已成为相当普遍的计算机外围设备，与 FFT 有关的软件也已成为很多软件平台中必备的工具。

本章先给出离散傅里叶变换的定义及其快速算法的 MATLAB 函数，在此基础上，介绍离散傅里叶变换在频谱分析和卷积计算中的应用。

14.1　离散傅里叶变换

离散傅里叶变换是对有限长序列 $x[n]$（其中 $0 \leqslant n \leqslant N-1$）定义的一种变

换，即

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\left(\frac{2\pi}{N}\right)kn} \quad (0 \leqslant k \leqslant N-1) \tag{14-1}$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\left(\frac{2\pi}{N}\right)kn} \quad (0 \leqslant n \leqslant N-1) \tag{14-2}$$

由于变换式在时域和频域均是离散的，因而适合于计算机编程计算。根据式(14-1)和式(14-2)，用 MATLAB 实现离散傅里叶正变换及逆变换的 M 文件如下：

```
function Xk = dft(x)
% Discrete Fourier Transform.
% Both x and Xk are row vectors.
[N,M] = size(x);
if M ~ = 1
    x = x.'; N = M;
end
Xk = zeros(1,N);
n = 0:N - 1;
for   k = 0:N - 1
        Xk(k + 1) = exp( - j * 2 * pi * k * n/N) * x;
end

function x = idft(Xk)
% Inverse Discrete Fourier Transform.
% Both x and Xk are row vectors.
[N,M] = size(Xk);
if M ~ = 1
    Xk = Xk.'; N = M;
end
x = zeros(1,N);
k = 0:N - 1;
for n = 0:N - 1
    x(n + 1) = exp(j * 2 * pi * k * n/N) * Xk;
end
```

```
x = x/N;
```

例 14-1 离散傅里叶变换

设 $x[n]=\{1,2,3,4\}$,求 $x[n]$ 的 4 点离散傅里叶变换,并用逆变换验证离散傅里叶变换的唯一性。

解 M 文件如下:

```
x = [1,2,3,4];
Xk = dft(x)
xn = idft(Xk)
```

程序运行结果为

```
Xk =
  10.0000   -2.0000 + 2.0000i   -2.0000 - 0.0000i   -2.0000 - 2.0000i
xn =
  1.0000 - 0.0000i   2.0000 - 0.0000i   3.0000 - 0.0000i   4.0000 + 0.0000i
```

14.2 离散傅里叶变换的快速算法

在信号处理中,DFT 的计算具有举足轻重的地位,信号的相关、滤波、谱估计等等都可以通过 DFT 来实现。然而,由 DFT 的变换式可以看出,求一个 N 点的 DFT 要 N^2 次复数乘法和 $N(N-1)$ 次复数加法。当 N 很大时,其计算量是相当可观的。

离散傅里叶变换(DFT)可以用快速傅里叶变换(FFT)的算法来计算。在 MATLAB 信号处理工具箱中,求解离散傅里叶变换的函数为 fft,求解逆离散傅里叶变换的函数为 ifft。此外,MATLAB 还提供了一个调整 fft 输出数据顺序的函数 fftshift,它们的调用格式如下:

```
Xk = fft(x)
Xk = fft(x,N)
```

表示计算信号 x 的快速离散傅里叶变换 Xk。当 x 为矩阵时,计算 x 中每一列信号的傅里叶变换。当 x 的长度 N 为 2 的整数次方时,采用基 2 算法,否则采用较慢的分裂基算法。当 length(x) > N 时,截断 x,否则补零。

```
x = ifft(Xk)
x = ifft(Xk,N)
```

表示计算 Xk 的逆离散傅里叶变换。当 Xk 为矩阵时,计算 Xk 中每一列逆变换。

X = fftshift(Xk)

表示调整 Xk 的顺序，将 Xk 中的左右两部分交换。当 Xk 为矩阵时，将 x 的左上、右上和右上、左下四部分两两交换。

例 14－2 快速傅里叶变换

用 fft 函数计算 $x[n]=\{1,2,3,4\}$ 的 4 点离散傅里叶变换。

解 M 文件如下：

```
x = [1,2,3,4];
Xk = fft(x)
xn = ifft(Xk)
```

输出结果如下：

```
Xk =  10    -2 + 2i    -2    -2 - 2i
xn =   1     2     3     4
```

14.3 离散傅里叶变换在信号频谱分析中的应用

1. 周期离散时间信号的谐波分析

周期为 N 的序列 $\tilde{x}[n]$，其傅里叶级数为

$$\tilde{x}[n]=\sum_{k=0}^{N-1}\tilde{X}[k]e^{j\left(\frac{2\pi}{N}\right)kn} \tag{14-3}$$

$$\tilde{X}[k]=\frac{1}{N}\sum_{n=0}^{N-1}\tilde{x}[n]e^{-j\left(\frac{2\pi}{N}\right)kn} \tag{14-4}$$

比较式(14－3)、(14－4)与式(14－1)、(14－2)可看出，如果只考虑 $\tilde{x}[n]$ 和 $\tilde{X}[k]$ 在主值区间的值，除因子 $\frac{1}{N}$ 所处的位置外，两组式子的形式相同，即

$$\tilde{X}[k]R[k]=\frac{1}{N}\cdot \mathrm{DFT}\{\tilde{x}[n]R[n]\} \tag{14-5}$$

$$\tilde{x}[n]R[n]=N\cdot \mathrm{IDFT}\{\tilde{X}[k]R[k]\} \tag{14-6}$$

其中，$R[n]$ 为从 0 开始、长度为 N 的单位矩形脉冲函数，即

$$R[n]=\begin{cases}1 & 0\leqslant n\leqslant N-1\\ 0 & 其余\ n\end{cases}$$

因此，计算离散傅里叶级数主值区间值的语句为

```
Xk = fft(x)/N
```

```
x = ifft(Xk) * N
```

其中,x、Xk 分别表示 $\tilde{x}[n]$、$\tilde{X}[k]$主区间值的向量。

例 14-3 周期离散时间信号的谐波分析

设周期序列 $\tilde{x}[n]$在主值区间的值 $\tilde{x}[n]R[n]=\{1,2,3,4\}$,求 $\tilde{x}[n]$的傅里叶系数 $\tilde{X}[k]$。

解 M 文件如下:

```
x = [1,2,3,4];
N = length(x);
Xk = fft(x)/N
```

输出结果为

```
Xk  =  2.5     -0.5 + 0.5i     -0.5     -0.5 - 0.5i
```

2. 非周期离散时间信号的频谱分析

离散时间傅里叶变换的公式为

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]\mathrm{e}^{-\mathrm{j}\Omega n} \tag{14-7}$$

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\Omega)\mathrm{e}^{\mathrm{j}\Omega n}\mathrm{d}\Omega \tag{14-8}$$

其中,$X(\Omega)$以 2π 为周期。

设 $x[n]$起始于 $n=0$,长度为 N,如果对 Ω 离散化,在一个周期中也取 N 个离散频率,令 $\Omega_k=2\pi k/N$,$X[k]=X(\Omega_k)$,则

$$X[k]R[k] = \sum_{n=0}^{N-1} x[n]\mathrm{e}^{-\mathrm{j}\left(\frac{2\pi}{N}\right)kn} \tag{14-9}$$

上式说明:$X(\Omega)$在主值区间$(0,2\pi)$的样本能够用离散傅里叶变换计算,即

$$X[k]R[k]=\mathrm{DFT}\{x[n]\}$$

按该算法在计算出 $X[k]R[k]=\mathrm{DFT}\{x[n]\}$后,对其做周期延拓,即可得到频谱的样本。

与式(14-9)对应的编程语句为

```
Xk = fft(x)
```

其中,x 和 Xk 分别表示 $x[n]$和 $X[k]R[k]$的向量。

在绘制频谱 $X(\Omega)$时注意,横坐标应该使用 Ω。

例 14-4 非周期离散时间信号的频谱分析

设 $x[n]=(0.8)^n$,$0\leqslant n\leqslant 31$,求 $x[n]$的离散时间傅里叶变换。

解 M 文件如下:

```
N = 32;
n = 0:N - 1;
x = 0.8.^n;
Xk = fft(x);
k = 0:N - 1;
F = k/N;
plot(F,abs(Xk));
xlabel('\Omega/(2\pi)');
ylabel('Magnitude')
```

在主值区间的幅度谱如图 14－1 所示。

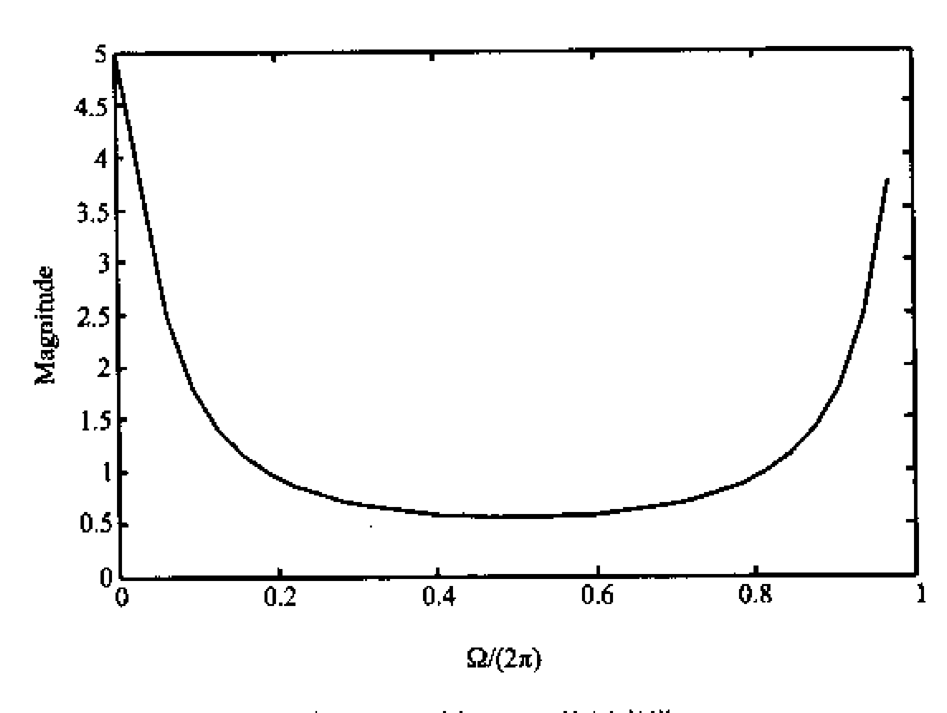


图 14－1　例 14－4 的幅度谱

3．周期连续时间信号的谐波分析

周期信号 $X(t)$ 的傅里叶级数表示式为

$$\tilde{x}(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\frac{2\pi}{T}t} \tag{14-10}$$

$$X_k = \frac{1}{T}\int_0^T \tilde{x}(t)\, e^{-jk\frac{2\pi}{T}t}\, dt \tag{14-11}$$

若对 $\tilde{x}(t)$ 进行整周期取样，设一个周期中取样 N 个点，且 N 为偶数，$\tilde{x}(t)$ 的样本为 $\tilde{x}[n]$，由取样知识，$\tilde{x}[n]$ 的离散傅里叶变换为 X_k 以周期 N 的延拓。在没有

频谱混叠的情况下，有

$$X_k = \mathrm{DFS}\{\tilde{x}[n]\} \quad \left(0 \leqslant k \leqslant \frac{N}{2}-1\right) \tag{14-12}$$

由式(14－5)得

$$X_k = \frac{1}{N}\cdot \mathrm{DFT}\{\tilde{x}[n]R[n]\} \quad \left(0 \leqslant k \leqslant \frac{N}{2}-1\right) \tag{14-13}$$

利用 fftshift 函数可对 DFT$\{\tilde{x}[n]R[n]\}$ 的顺序进行调整，将 X_0 调整在中心位置。利用 DFT 分析连续时间周期信号频谱的编程语句如下：

```
X = fft(x);
Xk = fftshift(X)/N;
```

如果对 X_k 进行周期延拓时发生了频谱混叠，则按式(14－13)只能进行近似分析。

例 14－5　周期连续时间信号的谐波分析 1

求图 14－2 所示周期矩形脉冲信号 $x(t)$ 的频谱。已知脉宽 $\tau=\frac{1}{4}$s，周期 $T=1$s，脉冲高度 $A=4$。设取一个周期中的取样点数 $N=32$。

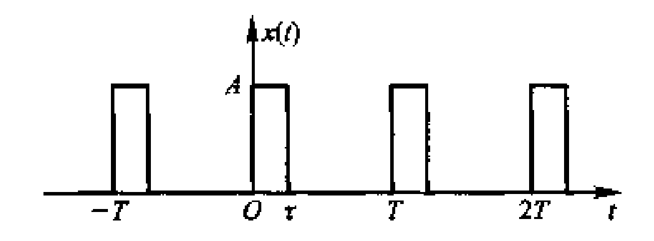


图 14－2　周期矩形脉冲信号

解　在信号不连续时刻取其中点值，主值区间的样本为

$$x[n]=\{\underset{n=0}{\underset{\uparrow}{2}},\underset{n=1}{\underset{\uparrow}{4}},\cdots,\underset{n=7}{\underset{\uparrow}{4}},\underset{n=8}{\underset{\uparrow}{2}},\underset{n=9}{\underset{\uparrow}{0}},\cdots,\underset{n=31}{\underset{\uparrow}{0}}\}$$

M 文件如下：

```
x = [2,4,4,4,4,4,4,4,2,zeros(1,23)];
N = 32;
X = fft(x);
Xk = fftshift(X)/N;
k = -N/2:N/2-1;
stem(k,abs(Xk),'filled');
```

```
xlabel('k'),ylabel('|Xk|')
```

程序运行结果如图 14-3 所示。

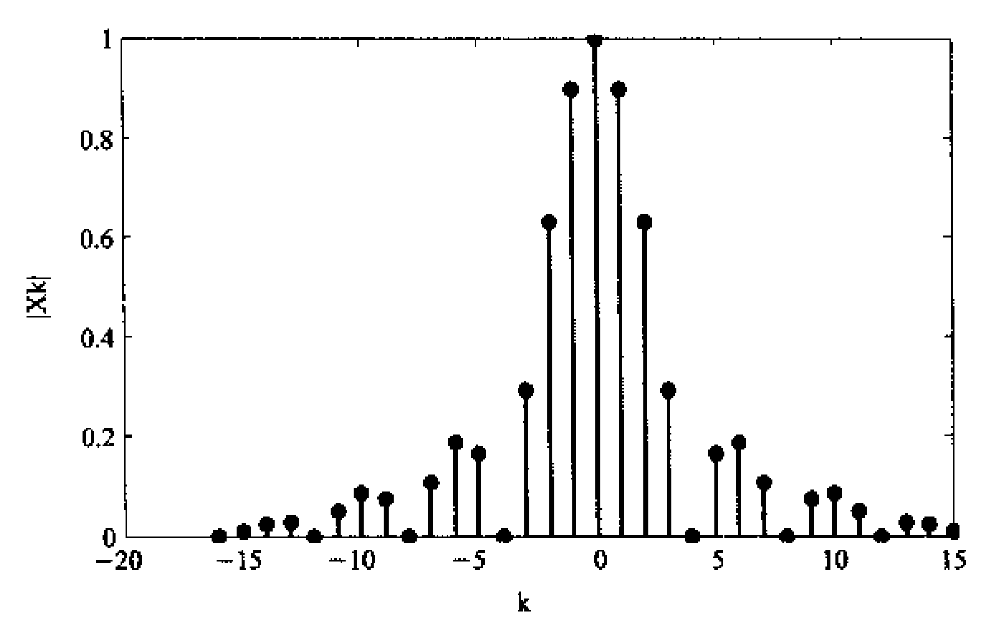


图 14-3 例 14-5 的频谱

例 14-6 周期连续时间信号的谐波分析 2

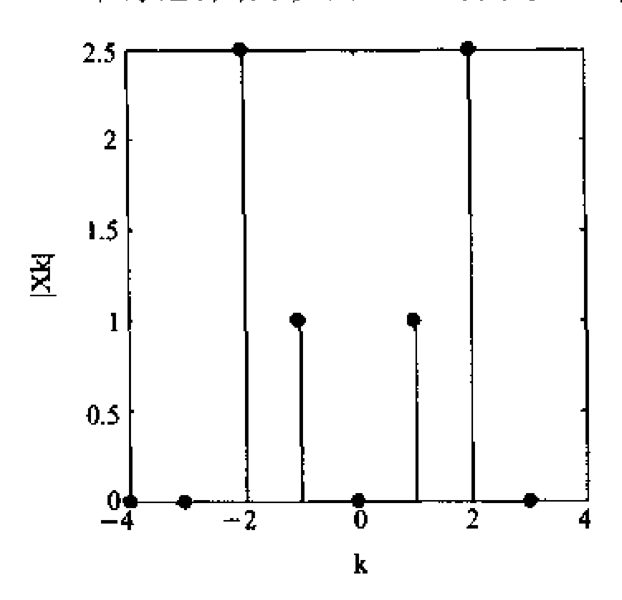
已知 $x(t)=2\sin(4\pi t)+5\cos(8\pi t)$，求傅里叶级数。

解 信号 $x(t)$ 的周期 $T=0.5\text{s}$，最高频率 $f_m=4\text{Hz}$，则取样间隔 $T_s<\frac{1}{2f_m}=0.125\ \text{s}$，一个周期中的取样点数 $N>\frac{T}{0.125}=4$，设取 $N=8$，M 文件如下：

```
N=8;T=0.5;Ts=T/N;
n=0:N-1;
t=n*Ts;
x=2*sin(4*pi*t)+5*cos(8*pi*t);
X=fft(x);
Xk=fftshift(X)/N;
k=-N/2:N/2-1;
subplot(1,2,1)
stem(k,abs(Xk),'filled');
xlabel('k'),ylabel('|Xk|')
Xkp=angle(Xk)*180/pi;
```

```
subplot(1,2,2)
stem(k,abs(Xkp),'filled');
xlabel('k'),ylabel('Phase(degrees)')
```

程序运行结果如图 14-4 所示。因 $X_0=0$,在相位频谱中,$k=0$ 的相位无意义。



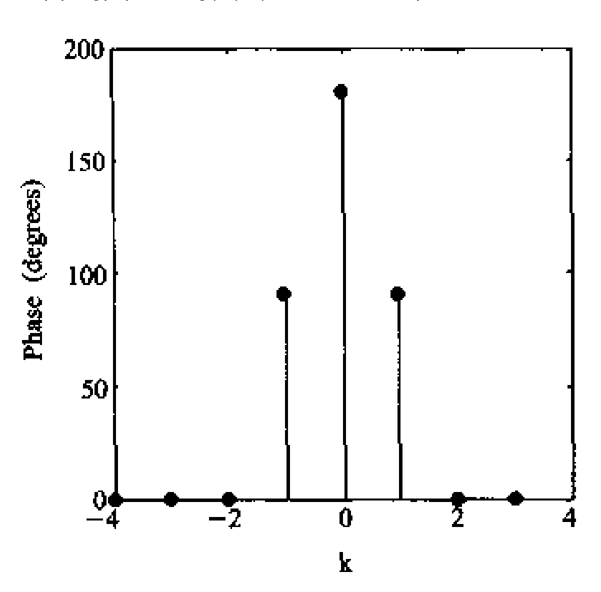


图 14-4　例 14-6 的频谱

4. 非周期连续时间信号的频谱分析

傅里叶变换对的公式为

$$X(\omega)=\int_{-\infty}^{\infty}x(t)e^{-j\omega t}dt \tag{14-14}$$

$$x(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}X(\omega)e^{j\omega t}d\omega \tag{14-15}$$

若对 $x(t)$ 取样,样本为 $x[n]$,则取样信号的频谱

$$X_s(\omega)=\frac{1}{T_s}\sum_{k=-\infty}^{\infty}X(\omega-k\omega_s)=\text{DTFT}\{x[n]\}$$

设 $X(\omega)$ 在周期延拓时的频谱混叠可忽略不计,由上式得

$$X(\omega)\approx T_s\cdot\text{DTFT}\{x[n]\}\quad(|\omega|<\omega_s/2)$$

利用离散时间傅里叶变换的频谱分析方法,再对频率离散化,使用 DFT 也可对非周期连续时间信号的频谱进行分析。

例 14-7　非周期连续时间信号的频谱分析

用离散傅里叶变换分析图 14-5 所示信号的频谱。设分析的最高角频率 $\omega_m=30$ rad/s,取样点数 $N=1\ 024$。

解　截取信号的长度

$$T=\frac{N\pi}{\omega_m}$$

取样间隔

$$T_s=T/N$$

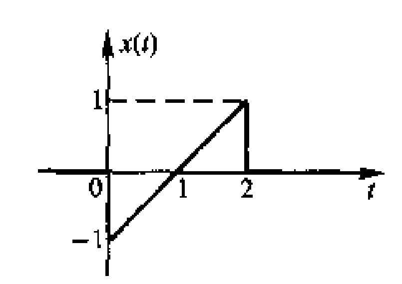


图 14－5 连续时间信号

M 文件如下：

```
N = input('Input N(even);');
wm = input('Input wm:');
T = pi * N/wm
Ts = T/N;   % time interval
Ws = 2 * pi/T; % omega interval
t = 0: Ts: 2;
x = [t - 1,zeros(1,N - length(t))];
Xk = fft(x);
X = fftshift(Xk) * Ts;
w = ( - N/2: N/2 - 1) * Ws;
plot(w,abs(X));
xlabel('w'),ylabel('|X(w)|')
```

运行程序时，输入 N＝1024，wm＝30，运行结果如图 14－6 所示。读者可用符号分析法求解图 14－5 所示信号的幅度频谱，与图 14－6 进行比较。

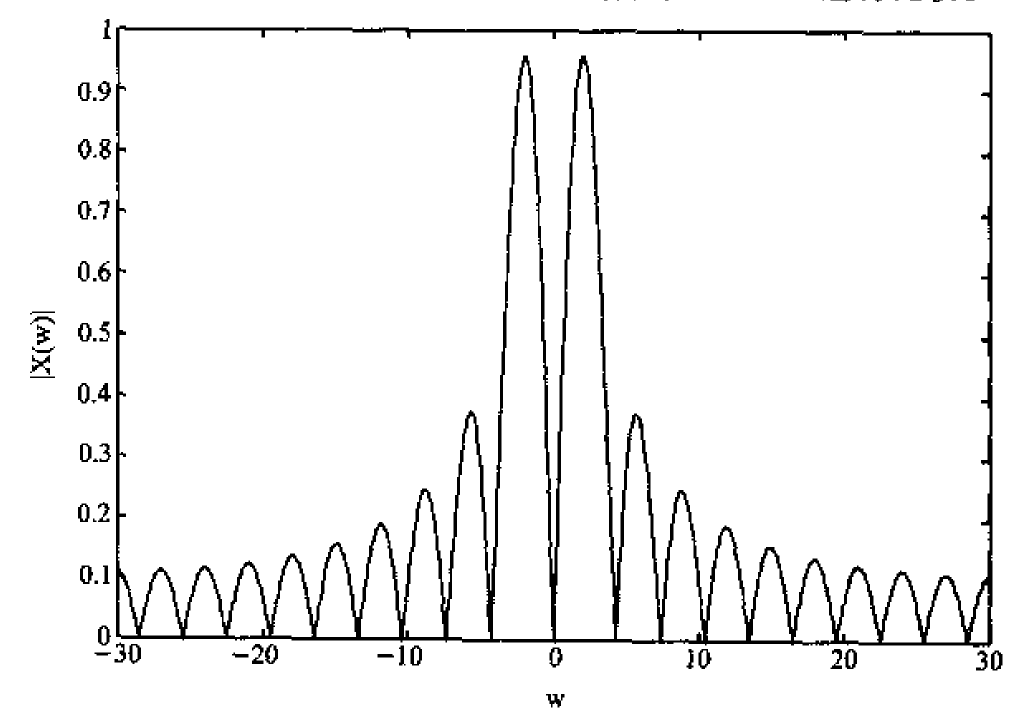


图 14－6 例 14－6 信号的幅度频谱

14.4 DFT在卷积计算中的应用

1. 圆周卷积

设$x[n]$和$h[n]$均为N点有限长序列,它们之间的N点圆周卷积定义为①

$$y[n]=\sum_{m=0}^{N-1}x[m]h[(n-m)_N]R_N[n] \tag{14-16}$$

为与线卷积相区别,圆周卷积用符号⊛表示②,即

$$y[n]=x[n]\circledast h[n]=h[n]\circledast x[n]$$

由定义可以看出,圆周卷积只在$0\leqslant m\leqslant N-1$区间内进行,圆周卷积结果也为$N$点有限长序列。对式(14-16),$h[(n-m)_N]$为$h[m]$的圆周反转$h[(-m)_N]$,再圆周移$n$位。因此,借助图形求解圆周卷积,其过程也包括变量代换、圆周反转、圆周移位、相乘、求和共5个步骤。以4点圆周卷积为例,$y[n]$与$x[n]$的关系为

$$\begin{bmatrix}y[0]\\y[1]\\y[2]\\y[3]\end{bmatrix}=\begin{bmatrix}h[0]&h[3]&h[2]&h[1]\\h[1]&h[0]&h[3]&h[2]\\h[2]&h[1]&h[0]&h[3]\\h[3]&h[2]&h[1]&h[0]\end{bmatrix}\begin{bmatrix}x[0]\\x[1]\\x[2]\\x[3]\end{bmatrix}$$

实现圆周卷积的函数circonv如下:

```
function y = circonv(x1,x2,N)
if length(x1) > N
   error('N必须> =x1的长度')
end
if length(x2) > N
   error('N必须> =x2的长度')
end
x1 = [x1,zeros(1,N - length(x1))];
x2 = [x2,zeros(1,N - length(x2))];
m = [0:1:N - 1];
```

① $h[(n-m)_N]$在其他书中也表示为$h[((n-m))_N]$、$h[n-m,\mathrm{mod}\ N]$或$h((n-m))_N$。

② N点圆周卷积也用符号Ⓝ表示,如4点圆周卷积表示为④。

```
x2 = x2(mod( - m,N) + 1);
H = zeros(N,N);
for n = 1:N
   H(n,:) = cirshift(x2,n - 1,N);
end
y = x1 * H.';
```

其中,cirshift 为实现 N 点圆周周移位的函数,圆周移 m 位的函数如下:

```
function y = cirshift(x,m,N)
if length(x) > N
   error('N 必须 > = x 的长度');
end
x = [x,zeros(1,N - length(x))];
n = [0:1:N - 1];
n = mod(n - m,N);
y = x(n + 1);
```

例 14-8 圆周卷积

设 $x_1[n]=\{1,2,2\}$, $x_2[n]=\{1,2,3,4\}$,计算 4 点圆周卷积 $x_1[n]\circledast x_2[n]$。

解 由于 $x_1[n]$的长度为 3,进行圆周卷积之前必须在其尾部增填一个零,使其成为 4 点序列。程序如下:

```
x1 = [1,2,2,0];
x2 = [1,2,3,4];
y = circonv(x1,x2,4)
```

结果为

$$x_1[n]\circledast x_2[n]=\{15,12,9,14\}$$

2. 用 DFT 计算线卷积

尽管圆周卷积与线卷积不是等同的概念,但如果在序列 $x[n]$、$h[n]$后都适当地补一些零值,以扩展其长度,那么,在作圆周卷积时,向右移去的零值,从左端出现仍取零值,这样就有可能得到与线卷积相同的结果。

设序列 $x[n]$和 $h[n]$是因果的,长度分别为 N_1 和 N_2,为方便起见,以下假定对这两个序列补零将长度调整为 N,其中 $N>\max(N_1,N_2)$。这两个序列的线卷积 $y_L[n]$和 N 点圆周卷积 $y_C[n]$之间的关系为

$$y_C[n] = \sum_{r=-\infty}^{\infty} y_L[n+rN]R_N[n] \tag{14-17}$$

式(14－17)说明，$y_C[n]$等于$y_L[n]$以N为周期延拓的主值序列。由于$y_L[n]$的长度为N_1+N_2-1，则当圆周卷积的长度

$$N \geqslant N_1 + N_2 - 1$$

时，$y_L[n]$的周期延拓不会发生混叠现象，则在该条件下圆周卷积与线卷积结果相等。

例 14－9　用圆周卷积计算线卷积

设$x_1[n]=\{1,2,2,1\}$，$x_2[n]=\{1,-1,1,-1\}$。(1)计算它们的线卷积$y_L[n]$；(2)计算7点圆周卷积$y_C[n]$。

解　由于$N\geqslant N_1+N_2-1$，圆周卷积应与线卷积相等。M文件如下：

```
x1 = [1,2,2,1];
x2 = [1, -1,1, -1];
yl = conv(x1,x2)
yc = circonv(x1,x2,7)
```

计算结果显示二者的确相等。

分析系统的响应实际上处理的是两个序列的线卷积问题。由线卷积的计算过程可知，完成线卷积共需N_1N_2次乘法运算。在$N_1=N_2=\frac{N+1}{2}$的情况下，即需要$\frac{1}{4}(N+1)^2$次乘法运算。当N的数值较大时，求卷积与直接进行DFT计算一样，也有实时性的问题。因此，实际应用中也可通过对序列补零求圆周卷积的方法计算线卷积，这主要因为圆周卷积可以借助快速傅里叶变换算法以较高的速度完成运算。

例 14－10　快速卷积

有一离散时间系统，其单位样值响应$h[n]=(0.8)^n\varepsilon[n]$，输入信号

$$x[n]=\begin{cases}1 & 0\leqslant n\leqslant 9\\ 0 & \text{其余 } n\end{cases}$$

试用$N=32$的DFT方法计算系统的输出$y[n]$。

解　先分别求得$h[n]$和$x[n]$的DFT，相乘后再求其逆变换。M文件如下：

```
N = 32;
n = 0:N - 1;
h = (0.8).^n;
```

```
Hk = fft(h,N);
x = ones(1,10);
Xk = fft(x,N);
Yk = Hk. * Xk;
y = abs(ifft(Yk,N));
stem(n,y,'filled');
```

程序运行结果如图 14 - 7 所示。

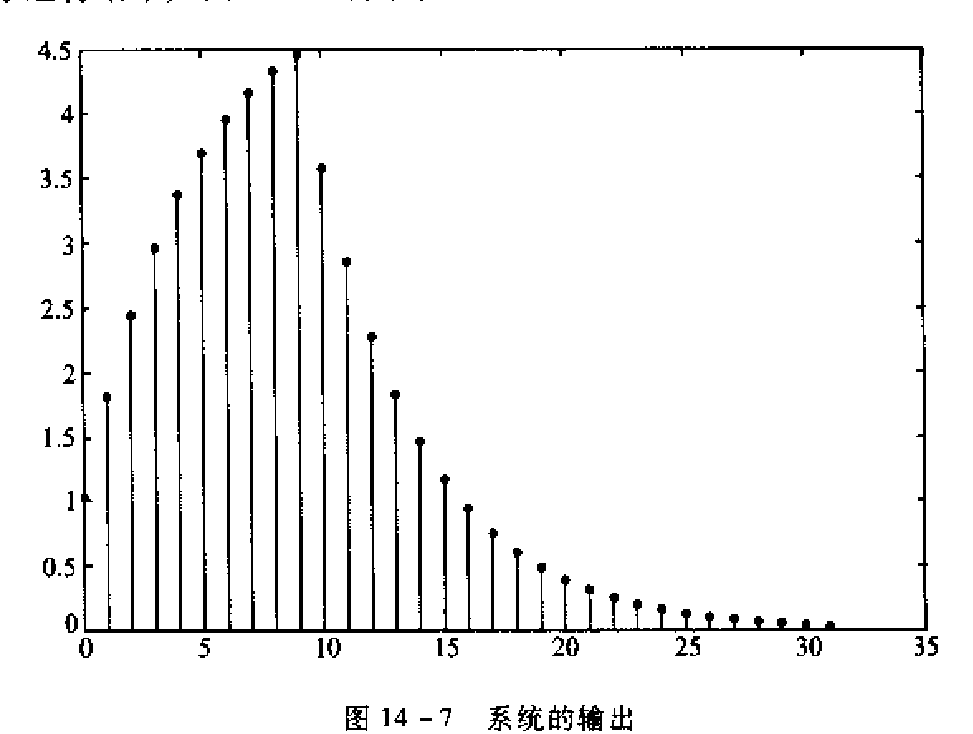


图 14 - 7　系统的输出

第十五章　模拟与数字滤波器

根据被处理的信号是模拟信号还是数字信号，滤波器分为模拟滤波器和数字滤波器。滤波器一般是在频域对信号进行加权运算的线性系统，这类系统具有频率选择功能，即对输入信号中所包含的各频率分量，系统允许某些分量通过，这些分量的增益相对比较大；而对另外一些分量，系统基本上不允许它们通过，它们的输出幅度和输入幅度相比变得很小。以离散时间系统为例，对于具有单位样值响应 $h[n]$ 的线性时不变因果系统，其输入输出关系为

$$y[n]=h[n]*x[n] \tag{15-1}$$

如果 $h[n]$、$x[n]$ 的傅里叶变换存在，则在频域的输入输出关系为

$$Y(z)\Big|_{z=e^{j\Omega}}=H(z)X(z)\Big|_{z=e^{j\Omega}} \tag{15-2}$$

可见，通过对 $H(z)\Big|_{z=e^{j\Omega}}$ 的设计，就可以实现对输入信号的频率选择。本章着眼于介绍如何使用 MATLAB 设计满足技术指标要求的系统函数。

显然，在理想情况下，人们要求在允许通过的频带（通带）内，各频率分量完全不失真地通过，而在不允许通过的频带（阻带）内，各频率分量被完全抑制掉。因此，对这种理想数字滤波器，其频率响应可用下式表示

$$H(z)\Big|_{z=e^{j\Omega}}=\begin{cases}A_0e^{-j\Omega n_p} & \text{（有所有通带内）}\\ 0 & \text{（在所有阻带内）}\end{cases} \tag{15-3}$$

根据通带在频率轴上所处的相对位置，滤波器可分为低通（LP）、高通（HP）、带通（BP）和带阻（BS）等的滤波器。顾名思义，模拟低通滤波器指滤波器允许信号低频段部分通过，而对信号的高频部分进行抑制；模拟高通滤波器允许信号的高频部分通过，对信号的低频部分进行抑制。数字滤波器的频率响应曲线是频率的周期函数，在 $0<\Omega<\pi$ 的频率范围内，数字低通滤波器允许低于通带频率 Ω_p 的频段 $0<\Omega<\Omega_p$ 部分的信号通过，而对高于阻带频率 Ω_s 的频段 $\Omega_s<\Omega<\pi$ 部分的信号进行抑制。

数字滤波器有两大类：有限样值响应数字滤波器（FIR）和无限样值响应数字滤波器（IIR），它们的设计过程是不一样的。

本章首先介绍模拟滤波器的设计，然后介绍 IIR 滤波器的设计，它是以模拟滤波器设计为基础，利用某种变换将模拟滤波器数字化，最后介绍 FIR 数字滤波器的设计。

15.1 模拟低通滤波器的设计

理想模拟滤波器在实际中是不可能实现的，因为其单位冲激响应 $h(t)$ 是非因果的，而实际滤波器必须是稳定和物理可实现的，所以必须在一定准则下对理想滤波器做近似。

给定模拟低通滤波器的技术指标：通带（截止）角频率 ω_p、通带（最大）衰减 α_p、阻带（截止）角频率 ω_s、阻带（最小）衰减 α_s，希望设计一个低通滤波器，其系统函数

$$H(s)=\frac{b_0s^M+b_1s^{M-1}+\cdots+b_{M-1}s+b_M}{a_0s^N+a_1s^{N-1}+\cdots+a_{N-1}s+a_N} \tag{15-4}$$

幅频响应的分贝值 $20\log|H(\mathrm{j}\omega)|$ 在 ω_p、ω_s 处分别达到 α_p、α_s 的要求。

滤波器设计的首要任务是要确定满足技术指标要求的系统函数，系统函数必须是物理可实现的，因而要满足因果性和稳定性。由于 $H(s)\big|_{s=\mathrm{j}\omega}=H(\mathrm{j}\omega)$ 为复数

$$H(\mathrm{j}\omega)=|H(\mathrm{j}\omega)|\mathrm{e}^{\mathrm{j}\varphi(\omega)}$$

则有

$$|H(\mathrm{j}\omega)|^2=H(\mathrm{j}\omega)H(-\mathrm{j}\omega)=H(s)H(-s)\Big|_{s=\mathrm{j}\omega} \tag{15-5}$$

由于系统函数 $H(s)$ 是 s 的有理函数，则 $H(s)H(-s)$ 是 s^2 的有理函数，而 $|H(\mathrm{j}\omega)|^2$ 必为 ω^2 的有理函数。

在研究滤波器的逼近函数问题中，一般由 $|H(\mathrm{j}\omega)|^2$ 确定 $H(s)$。当满足滤波器技术指标要求的 $|H(\mathrm{j}\omega)|^2$ 给定后，将 $|H(\mathrm{j}\omega)|^2$ 表达式中的 ω^2 用 $-s^2$ 替换，得出与 $|H(\mathrm{j}\omega)|^2$ 对应的 $H(s)H(-s)$，而后求出 $H(s)H(-s)$ 的极点和零点，由于 $H(s)H(-s)$ 是 s^2 的有理函数，若 p 为 $H(s)H(-s)$ 的一个极点（或零点），则 $-p$ 也必然为极点（或零点），故 $H(s)H(-s)$ 的极点（或零点）在 s 平面上的分布对称于 $\mathrm{j}\omega$ 轴。由于系统函数必须满足稳定性条件，则必须选择 $H(s)H(-s)$ 位于左半 s 平面的极点为 $H(s)$ 的极点。

模拟滤波器常见的逼近函数有：

(1) Butterworth（巴特沃思）逼近。

该逼近函数在通带和阻带内具有单调衰减的幅频特性。Butterworth 低通滤波器幅度函数的平方用下式给出

$$|H(j\omega)|^2=\frac{1}{1+\omega^{2N}} \tag{15-6}$$

式中，通带频率取为 1 rad/s；N 是系统函数的阶次。由上式可计算出：$\omega=0$ 时，$|H(j0)|=1$；$\omega=1$ rad/s 时，$|H(j1)|=\frac{1}{\sqrt{2}}$，即在通带频率处衰减的分贝值为 $-20\log(1/\sqrt{2})\approx 3$ dB；$\omega=\infty$ 时，$|H(j\infty)|=0$。

MATLAB 产生 Butterworth 模拟低通滤波器的函数是 buttap，格式为

```
[z,p,k] = buttap(n)
```

其中，n 为滤波器的阶次；z、p、k 分别为滤波器的零点、极点和增益。

例 15－1 Butterworth 模拟低通滤波器

求解三阶 Butterworth 低通滤波器的系统函数，并绘制幅频响应曲线。

解 M 文件如下：

```
[z,p,k] = buttap(3);
[b,a] = zp2tf(z,p,k)
[h1,w1] = freqs(b,a);
semilogx(w1,abs(h1)),grid
```

输出结果为

```
b =
     0          0          0       1.0000
a =
   1.0000    2.0000    2.0000    1.0000
```

即

$$H(s)=\frac{1}{s^3+2s^2+2s+1}$$

幅频响应曲线如图 15－1 所示。

(2) Chebyshev（切比雪夫）I 型逼近。

Chebyshev I 型系统函数的幅频特性在通带内呈等波纹变化，在阻带内单调衰减。在相同阶次条件下，Chebyshev 滤波器比 Butterworth 滤波器具有更好的衰减特性。Chebyshev 低通滤波器幅度函数的平方为

$$|H(j\omega)|^2=\frac{1}{1+\varepsilon^2C_N^2(\omega)} \tag{15-7}$$

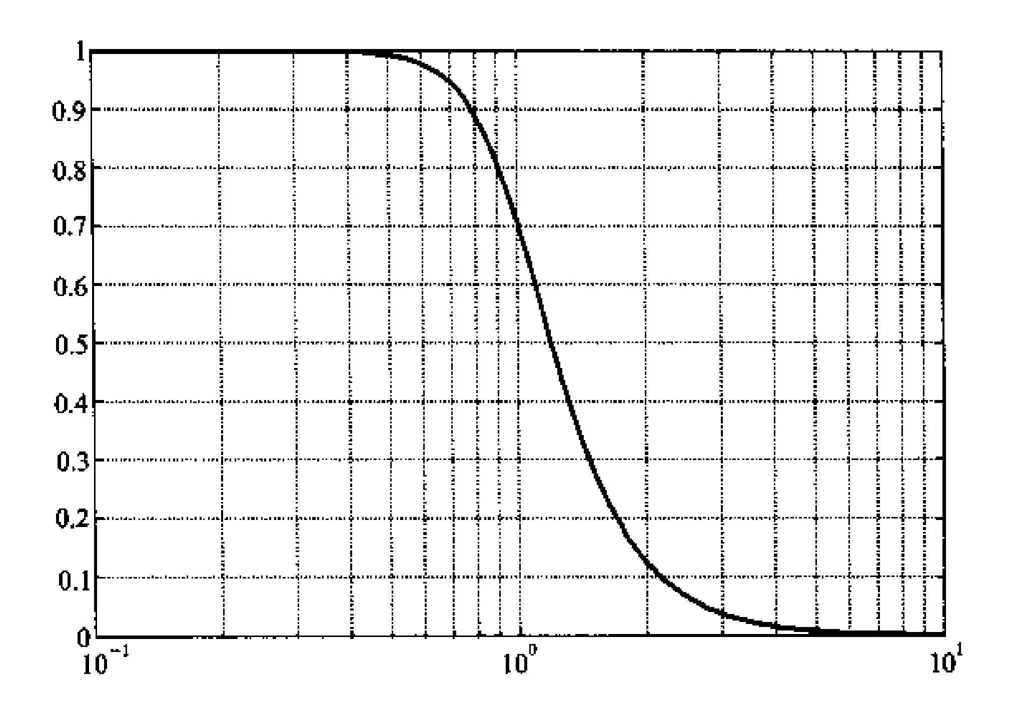


图 15－1 三阶 Butterworth 模拟低通滤波器的幅频响应

式中，$C_N(\omega)$是 N 阶 Chebyshev 多项式，定义为

$$C_N(\omega)=\cos[N\arccos\omega] \quad |\omega|\leqslant 1 \tag{15-8a}$$

$$C_N(\omega)=\cosh[N\,\mathrm{arcosh}\,\omega] \quad |\omega|>1 \tag{15-8b}$$

Chebyshev I 型模拟低通滤波器的指令格式为

```
[z,p,k]=cheb1ap(n,Rp)
```

其中，n 为滤波器的阶次；z、p、k 分别为滤波器的零点、极点和增益；Rp 是通带衰减的分贝值。

例 15－2 Chebyshev I 型模拟低通滤波器

绘制三阶 Chebyshev I 型模拟低通滤波器的频率响应，其中通带衰减为 3dB。

解 M 文件如下：

```
[z,p,k]=cheb1ap(3,3);
[b,a]=zp2tf(z,p,k);
[h1,w1]=freqs(b,a);
semilogx(w1,abs(h1)),grid
```

幅频响应如图 15－2 所示。

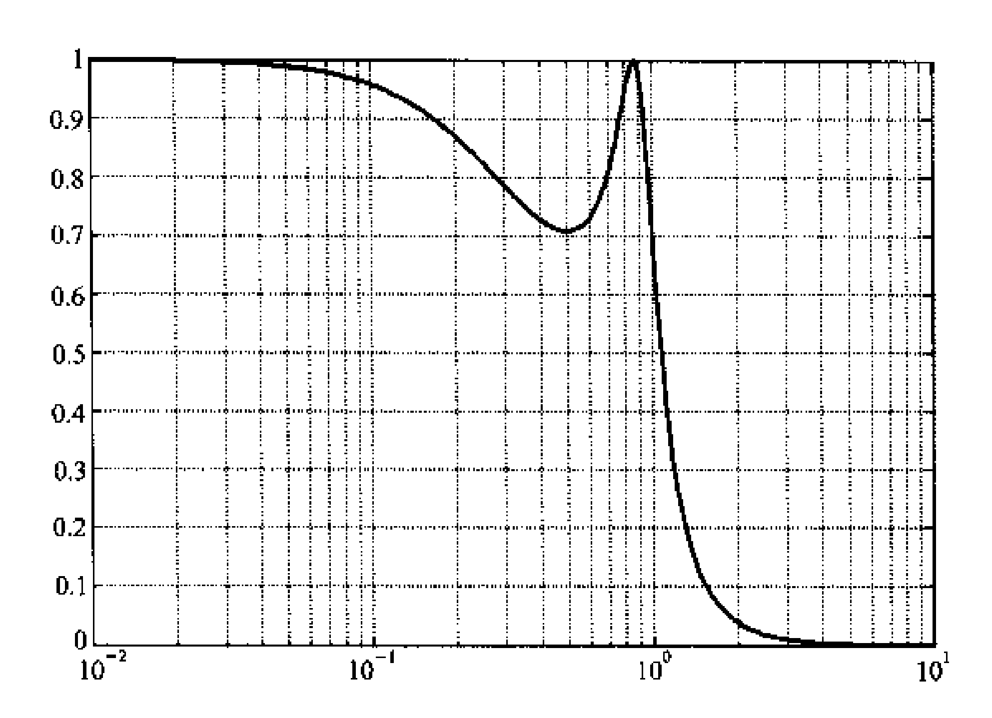


图 15－2 三阶 Chebyshev Ⅰ型模拟低通滤波器的幅频响应

(3) Chebyshev Ⅱ型逼近。

Chebyshev Ⅱ型逼近的幅频特性在通带内单调衰减，在阻带内等波纹变化。其幅度函数的平方为

$$|H(j\omega)|^2 = \frac{\varepsilon^2[C_N(1/\omega)]^2}{1+\varepsilon^2[C_N(1/\omega)]^2} \tag{15-9}$$

$C_N(\omega)$的定义见式(15－8)。

其指令格式为

[z,p,k] = cheb2ap(n,Rs)

其中，n 为滤波器的阶次；z、p、k 分别为传递函数的零点、极点、增益；Rs 为阻带衰减的分贝值。

例 15－3 Chebyshev Ⅱ型模拟低通滤波器

绘制四阶 Chebyshev Ⅱ型模拟低通滤波器的频率响应，要求幅度用分贝表示，阻带衰减大于 30dB。

解 M 文件如下：

```
[z,p,k] = cheb2ap(4,30);
[b,a] = zp2tf(z,p,k);
[h1,w1] = freqs(b,a);
semilogx(w1,20 * log10(abs(h1))),grid
```

幅频响应如图 15－3 所示。

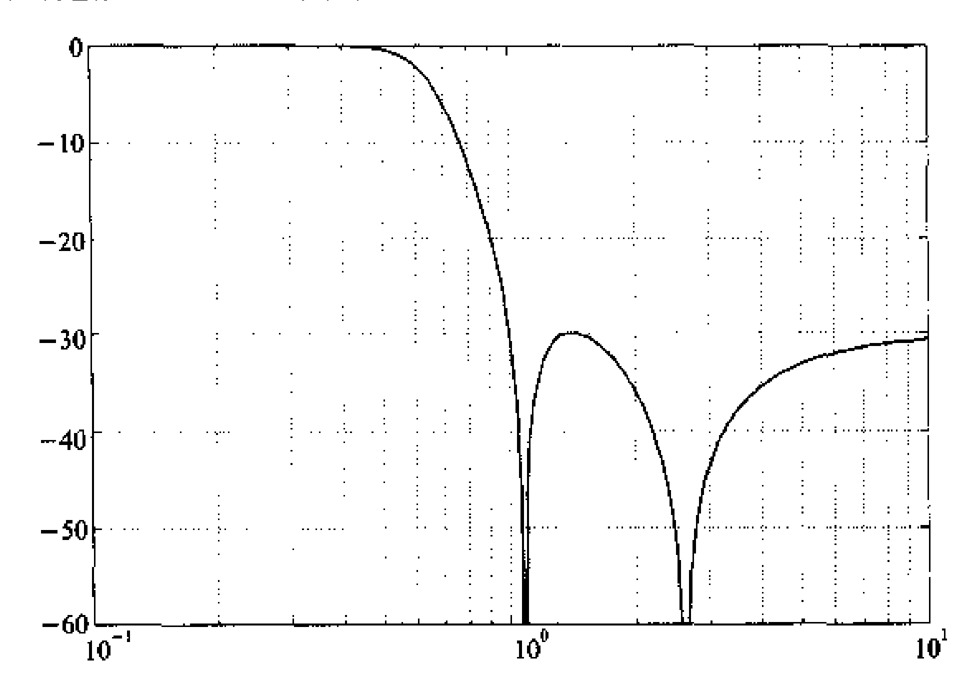


图 15－3 四阶 Chebyshev Ⅱ 型模拟低通滤波器的幅频响应

(4)椭圆逼近。

椭圆低通滤波器幅度函数的平方为

$$|H(j\omega)|^2=\frac{1}{1+\varepsilon^2U_n^2(\omega)} \tag{15-10}$$

其中，$U_n(\omega)$是雅可比椭圆函数。

椭圆滤波器的指令格式为

```
[z,p,k] = ellipap(n,Rp,Rs)
```

其中，Rp 为通带衰减的分贝值，Rs 为阻带衰减的分贝值。

例 15－4 椭圆模拟低通滤波器

绘制四阶椭圆模拟低通滤波器的频率响应，设通带衰减为 1 dB，阻带衰减为 40 dB。

解 M 文件如下：

```
[z,p,k] = ellipap(4,1,40);
[b,a] = zp2tf(z,p,k);
[h1,w1] = freqs(b,a);
semilogx(w1,abs(h1)),grid
```

幅频响应如图 15－4 所示。

(5)Bessel(贝塞尔)逼近。

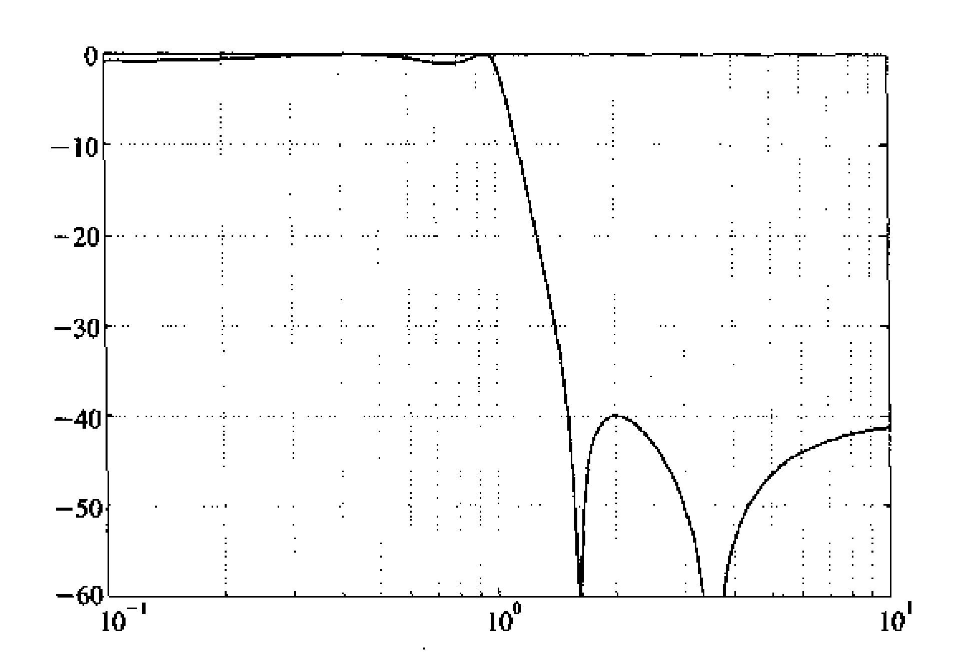


图 15－4 四阶椭圆模拟低通滤波器的幅频响应

其指令格式为

[z,p,k] = besselap(n)

当滤波器阶次 $n=1$ 时，3 dB 频率点为 1 rad/s；当阶次增加时，3 dB 所对应频率减小。

15.2 模拟滤波器的频率变换

借助模拟低通滤波器的系统函数，经合适的频率变换，可以得到高通、带通、带阻滤波器的系统函数。因此不论哪一种模拟滤波器的设计，都可以先将该滤波器的技术指标转化为角频率为 1rad/s 的低通滤波器的技术指标，按照该指标设计出低通滤波器的系统函数，再通过频率变换，得到所需类型系统函数。本节介绍 MATLAB 中实现频率变换的函数。

1. 低通到低通的变换

设通带角频率等于 1 rad/s 的低通滤波器的系统函数为 $H_1(s)$，通带角频率等于 ω_p 的低通滤波器的系统函数为 $H(s)$，低通到低通(lp2lp)的变换关系为

$$s \to \frac{s}{\omega_p} \quad 或 \quad H(s)=H_1\left(\frac{s}{\omega_p}\right)$$

其指令格式为

[bt,at] = lp2lp(b,a,wp)

[At,Bt,Ct,Dt] = lp2lp(A,B,C,D,wp)

其中,b、a 和 bt、at 分别为变换前和变换后的传递函数的分子和分母的系数,按 s 的降幂排列。第二种格式是状态空间形式的频率变换。

2. 低通到高通的变换

设高通滤波器的通带角频率为 ω_p,低通到高通(lp2hp)的变换关系为

$$s \rightarrow \frac{\omega_p}{s} \quad 或 \quad H(s) = H_l\left(\frac{\omega_p}{s}\right)$$

当低通滤器的角频率 ω 从 0 经 1 rad/s 到∞变化时,对应的高通滤波器的角频率 ω 则从 $-\infty$ 经 $-\omega_p$ 到 0 变化,由于幅频特性偶对称,则滤波器的通带由(0,1 rad/s)变换为(ω_p,∞)。

其指令格式为

[bt,at] = lp2hp(b,a,wp)

[At,Bt,Ct,Dt] = lp2hp(A,B,C,D,wp)

例 15-5 低通到高通的变换

设计通带频率为 200 Hz 的三阶 Butterworth 模拟高通滤波器。

解 M 文件如下:

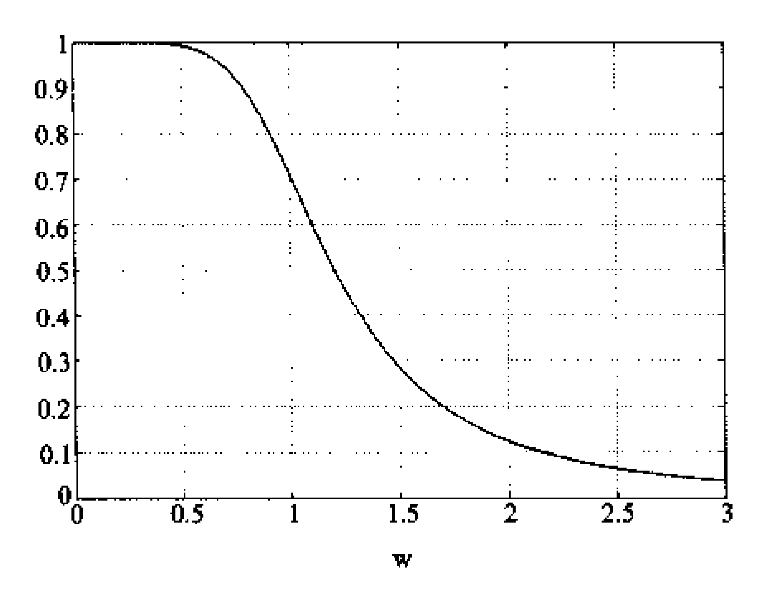
```
[z,p,k] = buttap(3);
[b,a] = zp2tf(z,p,k);
[bt,at] = lp2hp(b,a,200 * 2 * pi);
w1 = 0:0.01:3;
h1 = freqs(b,a,w1);
figure(1),plot(w1,abs(h1)),grid
xlabel('w')
f2 = 0:1:600;
h2 = freqs(bt,at,2 * pi * f2);
figure(2),plot(f2,abs(h2)),grid
xlabel('f')
```

频率特性如图 15-5 所示。

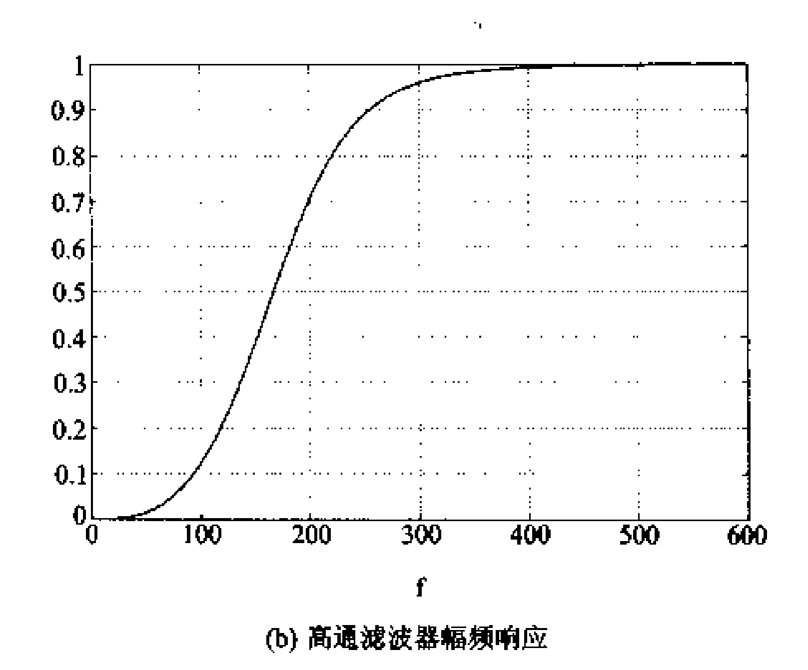
3. 低通到带通的变换

设带通滤波器的中心角频率为 ω_0,角频率带宽为 B,低通到带通(lp2bp)的变换关系为

$$s \rightarrow \frac{s^2 + \omega_0^2}{Bs}$$



(a) 低通滤波器幅频响应



(b) 高通滤波器幅频响应

图 15－5 通带频率为 200Hz 的模拟高通滤波器设计

其指令格式为

[bt,at] = lp2bp(b,a,w0,Bw)

[At,Bt,Ct,Dt] = lp2bp(A,B,C,D,w0,Bw)

该函数将通带频率为 1 rad/s 的模拟低通滤波器变换为带宽为 Bw、中心频率为 w0 的模拟带通滤波器。

例 15－6 低通到带通的变换

设计中心频率为 500 Hz、带宽为 400 Hz 的 6 阶 Butterworth 模拟带通滤波器。

解 M 文件如下：

```
[z,p,k] = buttap(3);
[b,a] = zp2tf(z,p,k);
[bt,at] = lp2bp(b,a,500 * 2 * pi,400 * 2 * pi);
f = linspace(0,1200,201);
h = freqs(bt,at,2 * pi * f);
plot (f,abs(h)),grid
xlabel('f')
```

频率响应如图 15－6 所示。

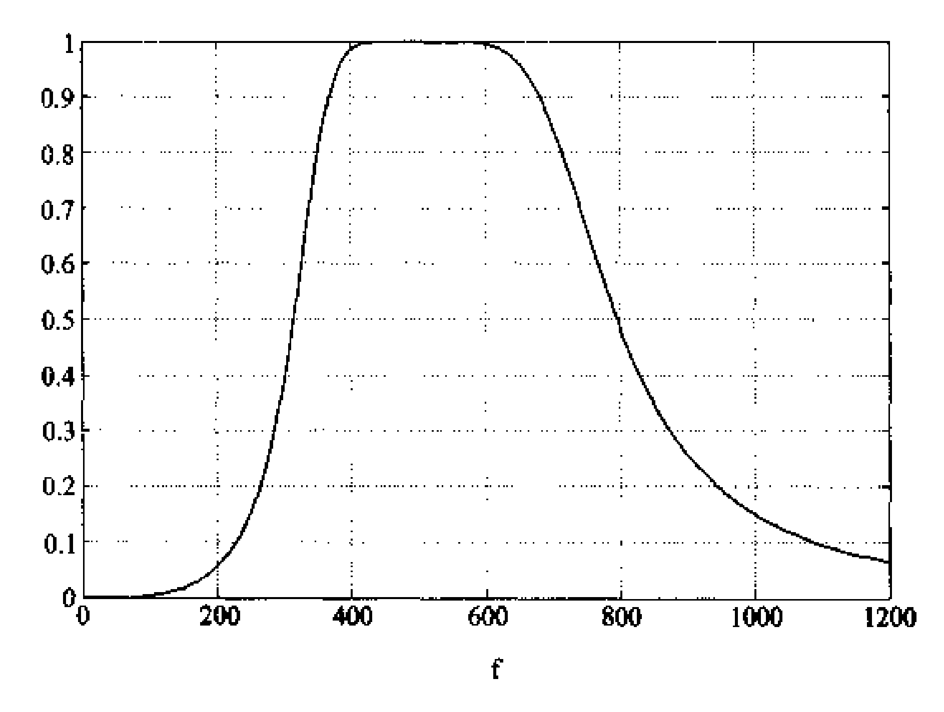


图 15－6 6 阶带通滤波器的频率响应

4. 低通到带阻的变换

设带阻滤波器的阻带中心频率为 ω_0，阻带带宽为 B，低通到带阻的变换关系为

$$s \to \frac{sB}{s^2 + \omega_0^2}$$

其指令格式为

```
[bt,at] = lp2bs(b,a,w0,Bw)
```

```
[At,Bt,Ct,Dt] = lp2bs(A,B,C,D,w0,Bw)
```

15.3 滤波器的最小阶次估计

给定滤波器的技术指标,一般希望用最小阶次的滤波器实现。MATLAB 信号处理工具箱提供了一组用于模拟/数字滤波器最小阶次估计的函数。

1. Butterworth 滤波器最小阶次估计

数字滤波器:[n,wn] = buttord(wp,ws,Rp,Rs)

模拟滤波器:[n,wn] = buttord(wp,ws,Rp,Rs,'s')

该函数根据给定的滤波器技术指标:通带频率 wp、阻带频率 ws、通带衰减和阻带衰减的分贝值 Rp 和 Rs,可以确定数字 Butterworth 低通、高通、带通和带阻滤波器的最小阶次 n,同时返回实际的截止频率 wn,'s'表示设计模拟滤波器。对数字滤波器,wp 和 ws 的取值范围为 0 < (wp,ws) < 1,1 对应于半取样频率。

若 wp 和 ws 均为标量,当 wp < ws 时指低通滤波器;当 wp > ws 时指高通滤波器;当 wp、ws 为二元向量时,若 ws(1) < wp(1) < wp(2) < ws(2),则为带通滤波器,若 wp(1) < ws(1) < ws(2) < wp(2),则为带阻滤波器。

例 15-7 Butterworth 滤波器最小阶次估计

设计一个 Butterworth 模拟低通滤波器满足下列条件:通带频率 800 Hz,通带衰减 3 dB,阻带频率 1 600 Hz,阻带衰减 30 dB。

解 M 文件如下:

```
[n,wn] = buttord(2 * pi * 800, 2 * pi * 1600,3,30,'s')
[b,a] = butter(n,wn,'s');
f1 = linspace(0,2000,201);
h1 = freqs(b,a,2 * pi * f1);
plot(f1,20 * log10(abs(h1))),grid
```

其中,butter()为求解给定阶次 Butterworth 滤波器传递函数、极零点等的函数,详见 15.5 节。

2. Chebyshev I 型滤波器最小阶次估计

数字滤波器:[n,wn] = cheb1ord(wp,ws,Rp,Rs)

模拟滤波器:[n,wn] = cheb1ord(wp,ws,Rp,Rs,'s')

例 15-8 Chebyshev I 型滤波器最小阶次估计

设计一个 Chebyshev I 型模拟带通滤波器满足下列条件:通带衰减 3 dB,通带范围 800 ~ 1 600 Hz,阻带频率 400 Hz 和 2 000 Hz,阻带衰减 40 dB。

解 M文件如下：

```
wp = [800,1600] * 2 * pi;
ws = [400,2000] * 2 * pi;
[n,wn] = cheb1ord(wp,ws,3,40,'s')
[b,a] = cheby1(n,3,wn,'s');
f1 = logspace(2,4,500);
h1 = freqs(b,a,2 * pi * f1);
semilogx (f1,abs(h1)),grid
xlabel('f')
```

幅频响应如图 15-7 所示。

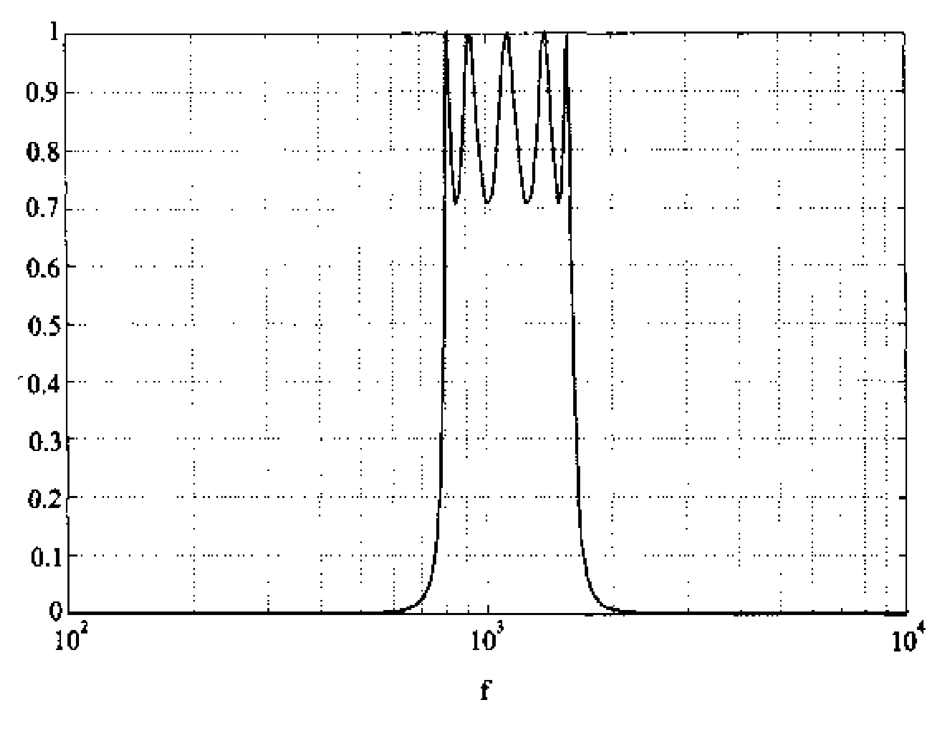


图 15-7 例 15-8 的幅频响应

3. Chebyshev Ⅱ型滤波器最小阶次估计

数字滤波器：[n,wn] = cheb2ord(wp,ws,Rp,Rs)

模拟滤波器：[n,wn] = cheb2ord(wp,ws,Rp,Rs,'s')

4. 椭圆滤波器最小阶次估计

数字滤波器：[n,wn] = ellipord(wp,ws,Rp,Rs)

模拟滤波器：[n,wn] = ellipord(wp,ws,Rp,Rs,'s')

15.4 IIR 滤波器的设计

本节介绍 IIR 数字滤波器设计的双线性变换法和冲激响应不变法。

1. 双线性变换法

对有限带宽的频谱进行映射，适当选取取样间隔 T 可以避免频谱混叠。模拟滤波器系统函数 $H_a(s)$ 的频谱具有无限宽度，为使映射后不发生频谱混叠，就必须将 $H_a(s)$ 的频谱从无限频率范围分布压缩至有限范围。若设新的频率变量为 ω_1，根据频谱不混叠的条件，以 ω_1 为变量的频谱必须分布在 $|\omega_1 T| < \pi$ 的频率范围内。实现频谱压缩的数学关系式为

$$\omega_1 T = 2\arctan(\omega/C) \tag{15-11}$$

式中，C 为正常数。当 ω 从 $-\infty$ 经 0 至 $+\infty$ 变化时，$\omega_1 T$ 则从 $-\pi$ 经 0 至 π 变化，式(15－11)为非线性单值映射，其逆关系则为

$$\omega = C\tan\left(\frac{\omega_1 T}{2}\right) \tag{15-12}$$

为了求出满足上式的复频域映射关系，设与 ω 和 ω_1 对应的复频率变量分别为 s 和 s_1，给式(15－12)两边同乘以 j，用 s 置换 $\mathrm{j}\omega$，用 s_1 置换 $\mathrm{j}\omega_1$，有

$$s = C\tanh\left(\frac{s_1 T}{2}\right) = C\frac{\mathrm{e}^{s_1 T/2} - \mathrm{e}^{-s_1 T/2}}{\mathrm{e}^{s_1 T/2} + \mathrm{e}^{-s_1 T/2}}$$

或

$$s = C\frac{\mathrm{e}^{s_1 T} - 1}{\mathrm{e}^{s_1 T} + 1} \tag{15-13}$$

式(15－13)显示，$H_a(s)$ 映射至 s_1 平面后，以 s_1 为变量的系统函数是 $\mathrm{e}^{s_1 T}$ 的有理函数，因而正好能使用标准映射关系 $z = \mathrm{e}^{s_1 T}$ 映射到 z 平面，从而有

$$s = C\frac{z-1}{z+1} \tag{15-14}$$

上式称之为双线性变换公式，它建立了 s 平面与 z 平面之间的变换关系，而且能够将有理 $H_a(s)$ 转换为有理 $H(z)$。

若用 Ω 表示数字角频率，与式(15－14)对应的频率关系式为

$$\omega = C\tan\left(\frac{\Omega}{2}\right) \tag{15-15}$$

式(15－15)对每一 ω 有唯一的 Ω 与之对应，因而数字滤波器的响应在形状上与模拟滤波器的一致，二者极值点的数目一定相同。双线性变换法不出现频率混

叠，但非线性关系却会导致数字滤波器的频率响应不能逼真地模仿模拟滤波器的频率响应。

MATLAB 中双线性变换的公式为

$$s = 2f_s \frac{z-1}{z+1} \tag{15-16}$$

$$s = \frac{2\pi f_p}{\tan(\pi f_p/f_s)} \frac{z-1}{z+1} \tag{15-17}$$

其中，fs 为取样频率，fp 为校正点模拟频率。其指令格式如下：

极零点增益模型：[zz,pz,kz] = bilinear(z,p,k,fs)

[zz,pz,kz] = bilinear(z,p,k,fs,fp)

传递函数模型： [bz,az] = bilinear(b,a,fs)

[bz,az] = bilinear(b,a,fs,fp)

状态空间模型： [Az,Bz,Cz,Dz] = bilinear(A,B,C,D,fs)

[Az,Bz,Cz,Dz] = bilinear(A,B,C,D,fs,fp)

为了减少数值误差，用双线性变换法设计数字滤波器最好直接对通带频率为 1rad/s 的模拟滤波器进行变换，其过程如下：

(1)将数字滤波器的技术指标转换为通带角频率为 1 rad/s 的模拟滤波器的技术指标，此时校正点频率 $f_{p1} = \frac{1}{2\pi}$Hz。

(2) 确定通带角频率为 1 rad/s 的模拟滤波器逼近函数和阶次。

(3) 计算校正点频率为 f_{p1}时的相对取样频率，公式为

$$f_{s1} = \frac{f_s f_{p1}}{f_p}$$

(4)利用 MATLAB 的双线性变换指令求数字滤波器的模型。以传递函数模型为例，指令为

```
[bz,az] = bilinear(b,a,fs1,1/(2*pi))
```

例 15-9 双线性变换法

设计一个取样频率为 8 kHz，通带频率为 2.4 kHz 的数字低通滤波器，使用四阶 Chebyshev Ⅱ 型模拟低通滤波器的频率响应，其阻带衰减 30 dB。

解 M 文件如下：

```
fs = 8000;fp = 2400;
[z,p,k] = cheb2ap(4,30);
[b,a] = zp2tf(z,p,k);
```

```
fp1 = 1/(2 * pi);
fs1 = fs * fp1/fp;
[bz,az] = bilinear(b,a,fs1,fp1);
[h1,f1] = freqz(bz,az,512,fs);
plot(f1,20 * log10(abs(h1))),grid
xlabel('f(Hz)'),ylabel('Magnitude(dB)')
```

数字滤波器的频率响应如图 15 - 8 所示。

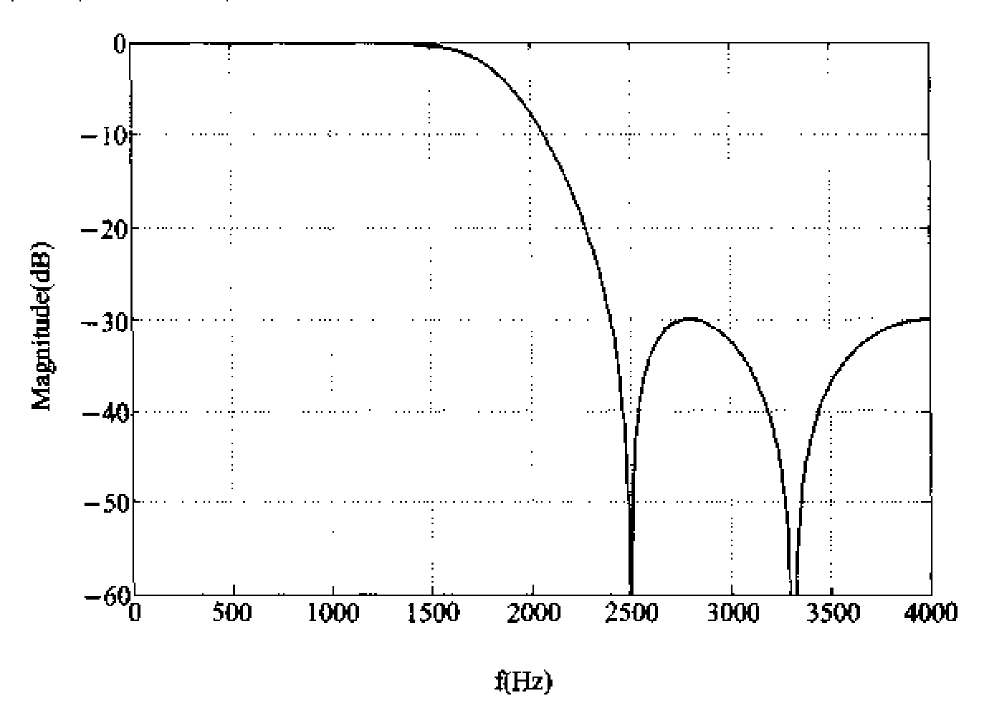


图 15 - 8 取样频率为 8 kHz,通带频率为 2.4 kHz 的数字低通滤波器的幅频响应

2. 冲激响应不变法

冲激响应不变法指数字滤波器的样值响应为模拟滤波器冲激响应的样本再乘以合适的因子。由于该方法不可避免的要发生频谱混叠现象,所以只适合设计低通和带通滤波器。根据冲激响应不变法的基本思想,可以按以下过程得到数字滤波器的系统函数:首先对模拟滤波器的系统函数 $H_a(s)$ 进行部分分式展开,对其取逆拉普拉斯变换得到冲激响应 $h_a(t)$,然后取 $h_a(t)$ 的取样值并乘以因子 K,构成数字滤波器的样值响应 $h[n]$,最后对 $h[n]$ 取 z 变换得到 $H(z)$。

使用冲激响应不变法设计数字滤波器的 MATLAB 函数是 impinvar,调用格式为

```
[bz,az] = impinvar(b,a,fs)
[bz,az] = impinvar(b,a,fs,TOL)
```

其中,fs 表示取样频率,单位为 Hz。如果不指定 fs,其值为 1 Hz。TOL 为误差容限,默认值是 0.000 1 。

例 15 - 10 冲激响应不变法

利用冲激响应不变法重新设计例 15 - 9 的数字低通滤波器。

解 M 文件如下:

```
fs = 8000; fp = 2400;
[z,p,k] = cheb2ap(4,30);
[b,a] = zp2tf(z,p,k);
[b,a] = lp2lp(b,a,2 * pi * fp);
[bz,az] = impinvar(b,a,fs);
[h1,f1] = freqz(bz,az,512,fs);
plot(f1,20 * log10(abs(h1))),grid
xlabel('f(Hz)'),ylabel('Magnitude(dB)')
```

幅频响应如图 15 - 9 所示,它与模拟滤波器的特性相差很大。

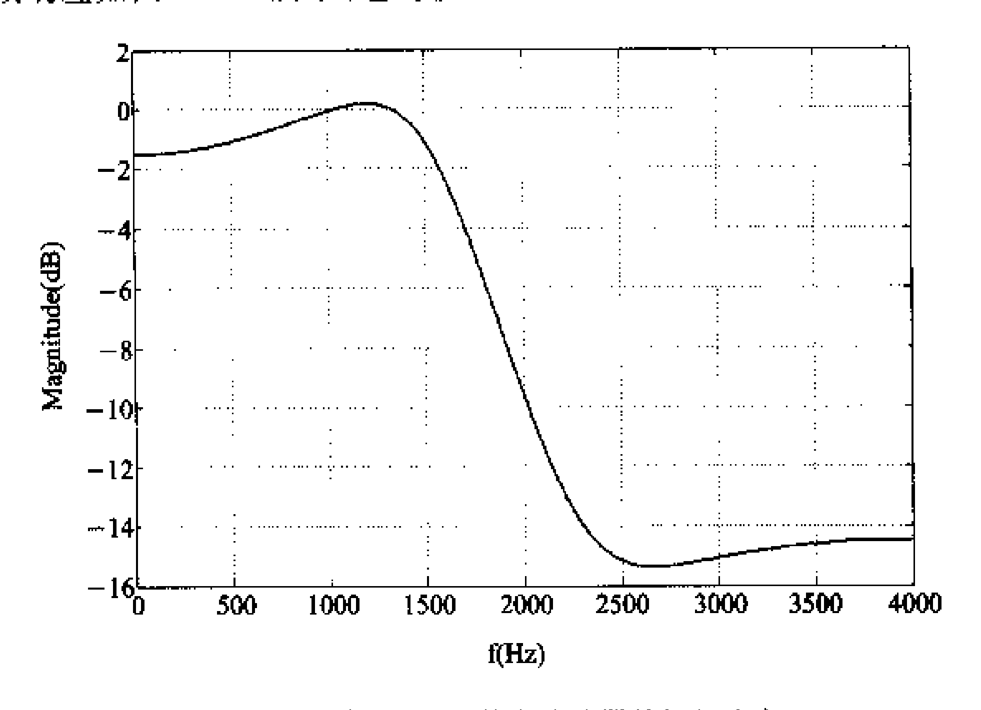


图 15 - 9 例 15 - 10 数字滤波器的幅频响应

15.5 MATLAB 标准滤波器的设计

利用模拟低通滤波器,通过适当的变换可以设计各种类型的数字滤波器。

但这样一来,设计滤波器的过程变得烦琐。MATLAB 提供了一组标准的模拟/数字滤波器设计函数,简化了模拟/数字滤波器的设计过程。

1. Butterworth 模拟/数字滤波器

[b,a] = butter(n,wn)

用于设计数字滤波器,其中 wn 在 0 至 1 范围取值,1 对应于半取样频率。当 wn 为标量时,设计阶次为 n 的低通滤波器。当 wn 为两个元素的向量时,设计阶次为 2 * n 的带通滤波器。

[b,a] = butter(n,wn,'ftype')

当 ftype 为 high 时,设计高通滤波器;当 ftype 为 stop 时,设计带阻滤波器,这时 wn 为二元向量,对应于阻带频率。

[b,a] = butter(n,wn,'s')

[b,a] = butter(n,wn,'ftype','s')

这两种格式用于设计模拟滤波器,wn 为模拟角频率。

例 15-11 Butterworth 数字带通滤波器

设取样频率为 8 000 Hz,设计一个 8 阶 Butterworth 数字带通滤波器,通带范围是 300 ~ 3 400 Hz。

解 M 文件如下:

```
wn = [300/4000,3400/4000];
[b,a] = butter(4,wn);
```

2. Chebyshev I 型模拟/数字滤波器设计

[b,a] = cheby1(n,Rp,wn)

[b,a] = cheby1(n,Rp,wn,'ftype')

[b,a] = cheby1(n,Rp,wn,'s')

[b,a] = cheby1(n,Rp,wn,'ftype', 's')

其中,Rp 为通带波纹的分贝值。

3. Chebyshev Ⅱ 型模拟/数字滤波器

[b,a] = cheby2(n,Rs,wn)

[b,a] = cheby2(n,Rs,wn,'ftype')

[b,a] = cheby2(n,Rs,wn,'s')

[b,a] = cheby2(n,Rs,wn,'ftype', 's')

其中,Rs 为阻带衰减的分贝值。

4. 椭圆模拟/数字滤波器

[b,a] = ellip(n,Rp,Rs,wn)

[b,a] = ellip(n,Rp,Rs,wn,'ftype')

[b,a] = ellip(n,Rp,Rs,wn,'s')

[b,a] = ellip(n,Rp,Rs,wn,'ftype', 's')

5. Bessel 模拟滤波器

[b,a] = besself(n,wn)

[b,a] = besself(n,wn,'ftype')

6. 递归数字滤波器

[b,a] = yulewalk(n,f,m)

yulewalk 函数采用对指定的频率响应进行最小均方拟合的方法设计递归的 IIR 数字滤波器。向量 f、m 指定了所希望的幅频响应形状。其中,f 为频率点向量,f = 1 时对应半取样频率。m 为对应的幅频响应向量,f、m 的长度必须相同。

15.6 FIR 滤波器的设计

IIR 滤波器的设计是以模拟滤波器的设计成果为基础的,其原因在于模拟滤波器本身就是无限冲激响应型的,即它的冲激响应在[0, +∞)上无限持续。

在相同的技术指标下,FIR 滤波器的阶次要比 IIR 滤波器的高,延迟时间也比较长。但是,FIR 滤波器也有它不可替代的优点,一是滤波器总是稳定的,二是容易实现线性相位,三是可以实现多带(通带或阻带)滤波器,四是易于硬件实现。

FIR 滤波器的设计建立在对理想滤波器频率特性的某种近似的基础上,不同的逼近方法构成了 FIR 滤波器的不同设计路线。

本节介绍 FIR 滤波器的窗函数设计法、最小平方逼近法和一致逼近法,以及相应的 MATLAB 函数。

1. 窗函数设计法

设所要设计的 FIR 滤波器的幅频响应为 $|H_d(e^{j\Omega})|$,相频响应 $\varphi(\Omega) = -M\Omega/2$,即具有线性相位,则滤波器的单位样值响应 $h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\Omega})e^{j\Omega}d\Omega$。一般情况下,$h_d[n]$是无限长序列,必须用一个有限长序列 $h[n]$ 去近似$h_d[n]$,也就是用窗函数将 $h_d[n]$截断。这种方法称为窗函数法。

FIR 滤波器的窗函数设计法不像 IIR 滤波器的设计那样能够精确指定通带、阻带的截止频率和波纹系数,而仅能大致给定,其他的参数是靠 $h[n]$的长度 M 及所使用的窗函数的性能来决定的。在选定窗函数后,可以不断变化 M,以

检查通带、阻带是否达到要求，直到满意为止。

值得一提的是，FIR 滤波器的窗函数设计法不仅可以设计标准频率响应（低通、高通、带通、带阻）滤波器，还可以设计任意频率响应的多带滤波器。

MATLAB 信号处理工具箱提供了两个用窗函数法设计 FIR 滤波器的函数，fir1 和 fir2。前者用于设计诸如低通、带通等的滤波器，后者用于设计多带滤波器。

```
b = fir1(n,fn)
b = fir1(n,fn,window)
```

返回 n 阶 FIR 滤波器，b 为滤波器的系数向量（按 z 的降幂排列），fn 为截止频率，与其他滤波器不同，此处截止频率定义为 -6 dB 处的频率，fn = 1 对应于半取样频率。当 fn 为标量时，设计低通滤波器；当 fn 为二元向量，即 fn = [f1,f2]（f1 < f2），设计带通滤波器。参数 windows 用于指定所选的窗函数种类，缺省的为 Hamming 窗。窗函数的长度为 n + 1，其他可以使用的窗还有：hann、bartlett、blackman、chebwin 等。

```
b = fir1(n,fn,'ftype')
b = fir1(n,fn,'ftype',window)
```

当 fn 为标量，ftype 为 high 时，设计高通滤波器；当 fn 为二元向量，ftype 为 stop 时，设计带阻滤波器。

例 15-12 设计 FIR 滤波器的窗函数法 1

用窗函数法设计 64 阶，通带数字角频率为 0.75π rad 的数字高通滤波器。

解 M 文件如下：

```
b = fir1(64,0.75,'high',chebwin(65,30))
freqz(b,1,512)
```

频率响应如图 15-10 所示。

用窗函数法设计频率响应具有分段线性特征 FIR 滤波器的函数为 fir2。

```
b = fir2(n,f,m)
b = fir2(n,f,m,window)
```

返回 n 阶 FIR 滤波器，其幅频响应由向量 f 和 m 指定。f 为频率点向量，$0 \leqslant f \leqslant 1$，$f = 1$ 对应半取样频率，m 为相应的幅频响应向量。f 和 m 的长度必须相同。b 为滤波器的系数向量（按 z 的降幂排列）。参数 window 用于指定所选的窗函数种类。

```
b = fir2(n,f,m,npt)
b = fir2(n,f,m,npt,window)
```

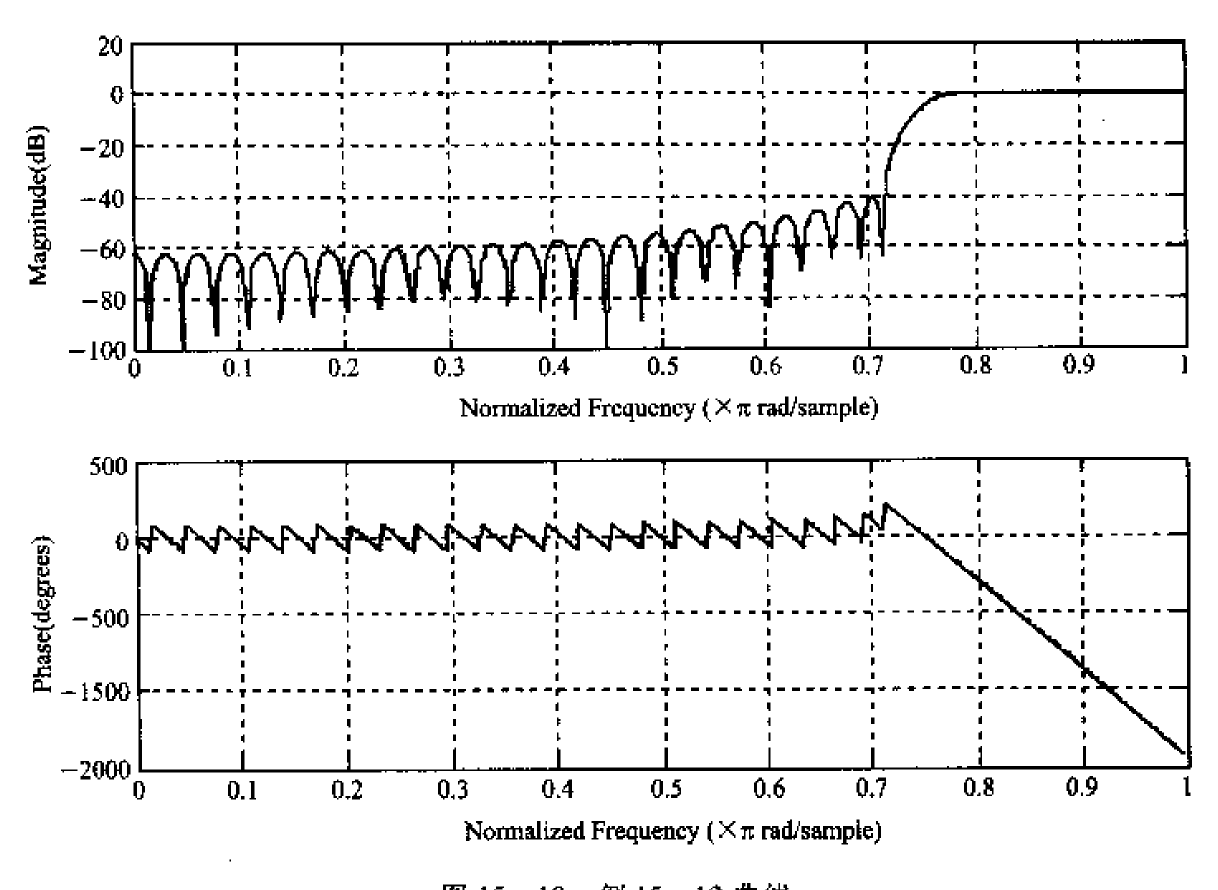


图 15-10 例 15-12 曲线

```
b = fir2(n,f,m,npt,lap)
b = fir2(n,f,m,npt,lap,window)
```

其中,参数 npt 指定对给定的幅频响应做 npt 点内插;参数 lap 指定在重复的频率点附近插入的区域大小。

例 15-13 设计 FIR 滤波器的窗函数法 2

理想幅频响应如图 15-11 中的折线所示,设计 30 阶 FIR 滤波器,并绘制幅频响应曲线。

解 M 文件如下:

```
f = [0,0.2,0.2,0.4,0.4,0.6,0.6,1];
m = [0,0,1,1,0,0,1,1];
b = fir2(30,f,m);
[h,w] = freqz(b,1,128);
plot(f,m,w/pi,abs(h))
grid;
```

幅频响应如图 15-11 所示。

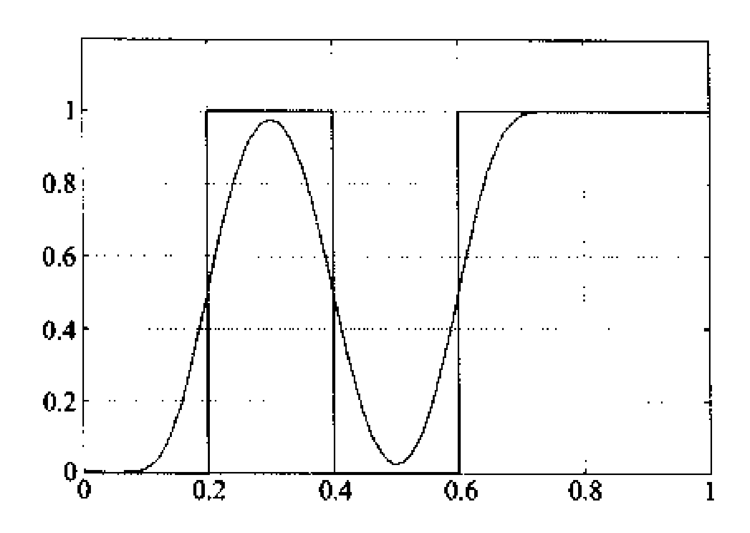


图 15－11 设计指定幅频响应的多带 FIR 滤波器

2. 最小平方法

对于给定的滤波器频率响应 $|H_d(e^{j\Omega})|$，所谓最小平方逼近，就是寻找一个 $H(e^{j\Omega})$ 去逼近 $H_d(e^{j\Omega})$，使得方差 $\int[H(e^{j\Omega})-H_d(e^{j\Omega})]^2\mathrm{d}\Omega$ 最小。这种设计方法着眼于区间上的总误差最小，但并不保证在每个局部位置误差最小。

用最小平方法设计线性相位 FIR 滤波器的函数如下：

```
b = firls(n,f,m)
```

返回一个长度为 n＋1 的线性相位 FIR 滤波器，其幅频响应由向量 f 和 m 指定。其中，0≤f≤1 为频率拐点向量，f＝1 对应半取样频率；m 为相应的滤波器幅度向量。f 和 m 的长度必须相同。b 为滤波器的系数向量，满足

$$b(k)=b(n+2-k),\quad k=1,2,\cdots,n+1$$

为第Ⅰ类（奇数 n）和第Ⅱ类（偶数 n）线性相位滤波器。

```
b = firls(n,f,m,'ftype')
```

参数 ftype 有两个选项：

Hilbert：设计第Ⅲ类和第Ⅳ类具有奇对称特性的线性相位滤波器，向量 b 满足

$$b(k)=-b(n+2-k),\quad k=1,2,\cdots,n+1$$

differentiator：设计的滤波器仍为奇对称的线性相位滤波器，但同时对误差做了加权，使低频段误差大大小于高频段误差。

例 15－14 设计 FIR 滤波器的最小平方法

设计一个 24 阶的 FIR 滤波器，其幅频响应具有分段线性通带，如图 15－12 所示。

解 M 文件如下：

```
f = [0,0.3,0.4,0.6,0.7,0.9];
m = [0,1,0.7,0.7,0.5,0.5];
b = firls(24,f,m,'differentiator')
[h,f1] = freqz(b,1,512,2);
plot(f1,abs(h),f,m),grid
```

输出曲线如图 15－12 所示。

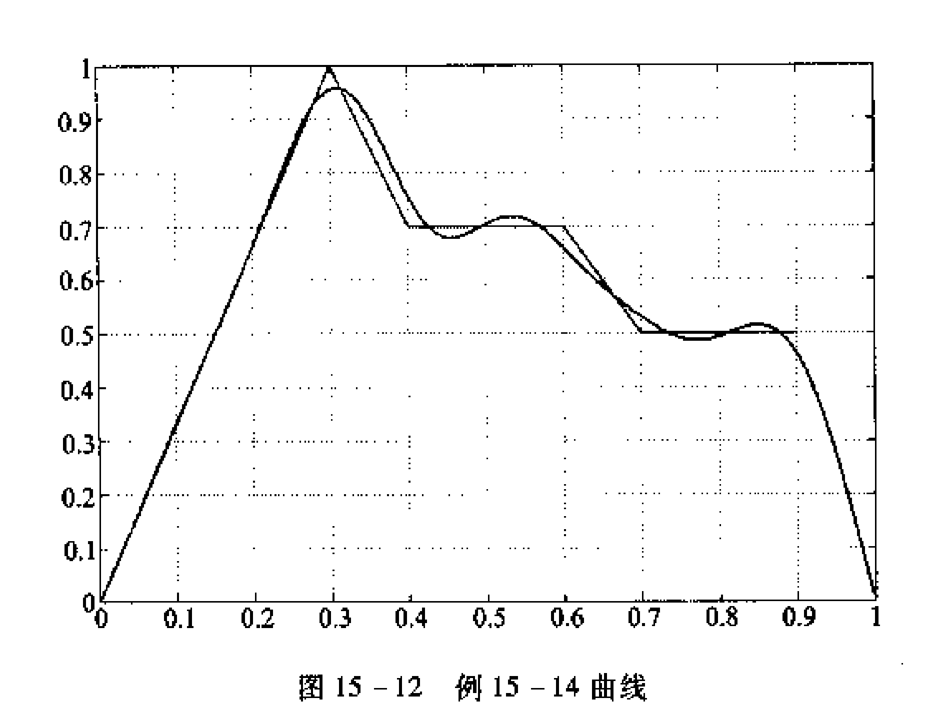


图 15－12　例 15－14 曲线

另一用最小平方法设计 FIR 线性相位滤波器的函数如下：

b = fircls(n,f,m,up,lo)

b = fircls(n,f,m,up,lo,'design_flag')

返回长度为 n + 1 的线性相位 FIR 滤波器，b 为滤波器的系数向量。幅频响应由向量 f 和 m 指定，f = 1 对应于半取样频率，m 是由滤波器幅频响应中各频带增益组成的向量，其长度等于 f 的长度减 1。up 和 lo 与 m 的长度相同，分别表示各频带所容许的最大和最小波动量。design_flag 用来指定函数返回结果的形式

trace：函数以文本的形式返回滤波器系数；

plots：函数画出滤波器及各个频带的幅频响应曲线、群迟延、零点和极点；

both：函数既以文本的形式返回滤波器系数，又画出滤波器及各个频带的幅频响应曲线、群迟延、零点和极点。

用最小平方法设计低通或高通线性相位 FIR 滤波器的函数为

```
b = fircls1(n,fn,Rp,Rs)
b = fircls1(n,fn,Rp,Rs,'high')
b = fircls1(n,fn,Rp,Rs,'design_flag')
```

Rp 和 Rs 分别表示通带和阻带的最大波纹。'high'指高通滤波器。

例 15-15 设计 FIR 低通滤波器的最小平方法

解 M 文件如下：

```
n=55;
fn=0.3;
Rp=0.02;
Rs=0.008;
b = fircls1(n,fn,Rp,Rs,'plots');
```

输出曲线如图 15-13 所示。

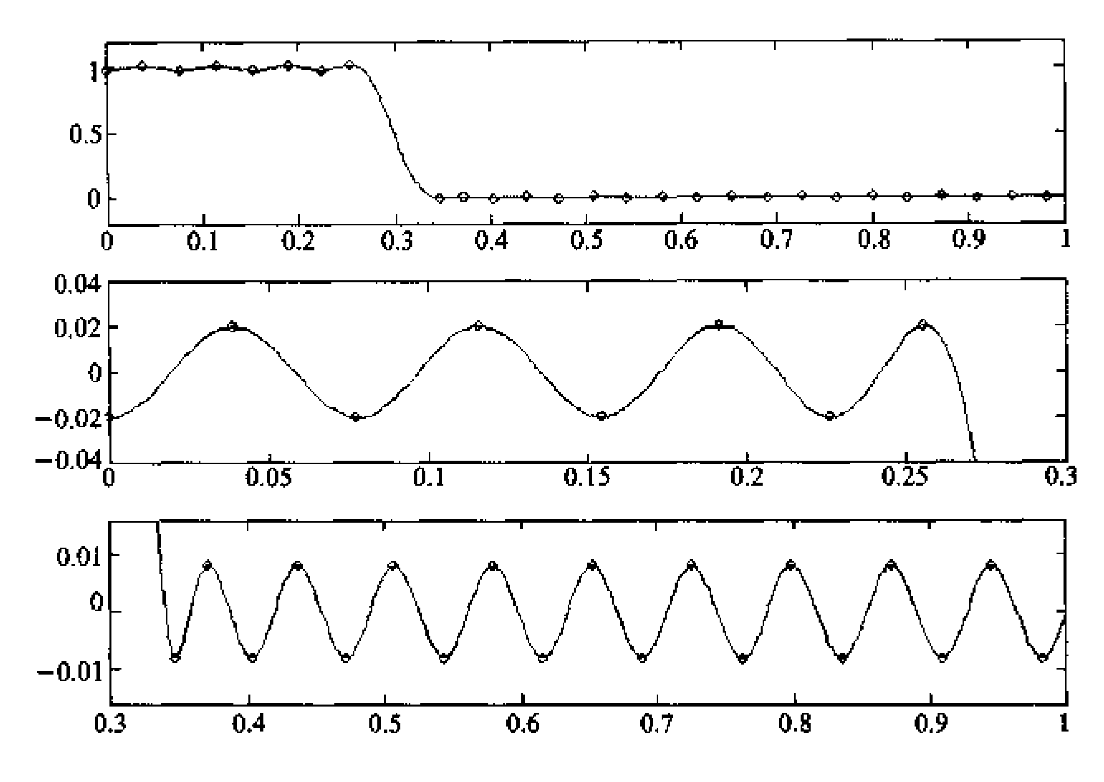


图 15-13 用最小平方法设计 FIR 低通滤波器

3. 最佳一致逼近法

给定滤波器的幅频响应 $|H_d(e^{j\Omega})|$，所谓一致逼近，就是寻找一个近似的 $H(e^{j\Omega})$，使得在整个频域上，$\max\{|H(e^{j\Omega})-H_d(e^{j\Omega})|\}$最小，这种逼近方法称为最佳一致逼近。Chebyshev 解决了最佳一致逼近的存在性、唯一性以及如何构造

等问题。

McClellan J. H、Parks T. W 和 Rabiner L. R 等人应用 Chebyshev 逼近理论，提出了一种 FIR 滤波器的设计方法，这种方法在一致意义上对 $H_d(e^{j\Omega})$ 作最佳一致逼近，可以获得较好的通带和阻带性能，并能精确地指定通带频率和阻带频率。

使用最佳一致逼近法设计 FIR 滤波器的函数为

```
b = remez(n,f,m)
b = remez(n,f,m,w)
b = remez(n,f,m,'ftype')
b = remez(n,f,m,w,'ftype')
```

其中，n 为 FIR 滤波器的阶次；其幅频响应由向量 f 和 m 指定，这两个向量的长度必须相同，而且为偶数；b 为滤波器的系数向量；权值向量 w 对各频段作加权拟合，每一频段给定一个值；参数 ftype 为 Hilbert 或 differentiator 。

确定滤波器最小阶次的函数为

```
[n,fo,mo,w] = remezord(f,m,dev)
[n,fo,mo,w] = remezord(f,m,dev,fs)
```

其中，dev 为实际幅频响应与给定幅频响应间最大容许的偏差，其向量长度与 m 相同；fs 为取样频率；n 为滤波器阶次；fo 和 mo 分别为函数返回的拐点频率和幅度的向量；w 为权值向量。

例 15－16 设计 FIR 滤波器的最佳一致逼近法

设计一个最小阶次 FIR 低通滤波器，通带频率为 400 Hz，阻带频率为 600 Hz，取样频率为 2 000 Hz，通带衰减小于 3 dB，阻带衰减大于 30 dB。

解 M 文件如下：

```
Rp = 3;
Rs = 30;
fs = 2000;
f = [400,600];
m = [1,0];
dev = [(10^(Rp/20) - 1)/(10^(Rp/20) + 1) ,10^( - Rs/20)];
[n,fo,mo,w] = remezord(f,m,dev,fs);
b = remez(n,fo,mo,w);
[h,f] = freqz(b,1,1024,fs);
plot(f,20 * log10(abs(h)))
```

grid

幅频响应如图 15 - 14 所示。

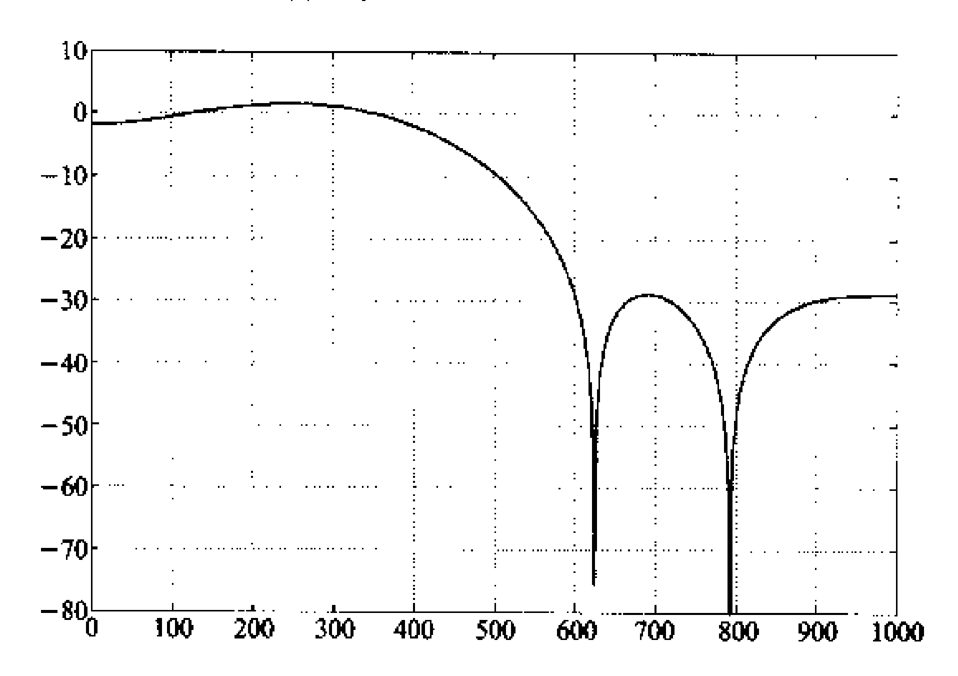


图 15 - 14 最小阶次 FIR 滤波器的幅频响应

附录

例题索引

第 二 篇

例题名 页码

第 三 篇

例题名 页码

例题名 页码

例题名 页码

参考书目

1 张志涌等编著.精通MATLAB5.3版.北京:北京航空航天大学出版社,2000

2 陈怀琛,吴大正,高西全编著.MATLAB及在电子信息课程中的应用.北京:电子工业出版社,2002

3 刘树棠译.《信号与系统》计算机练习——利用MATLAB.西安:西安交通大学出版社,2000

4 Kamen E W, Heck B S. 信号与系统基础.北京:科学出版社,培生教育出版集团,2002

5 吴新余,周井泉,沈元隆等编著.信号与系统——时域、频域分析及MATLAB软件应用.北京:电子工业出版社,1999

6 龚剑,朱亮编著.MATLAB5.x入门与提高.北京:清华大学出版社,2000

7 刘卫国主编.科学计算与语言.北京:中国铁道出版社,2000

8 导向科技.MATLAB6.0程序设计与实例应用.北京:中国铁道出版社,2001

9 张志涌,徐彦琴等编著.MATLAB教程——基于6.x版本.北京:北京航空航天大学出版社,2001

10 刘宏友,李莉,彭锋等编著.MATLAB 6基础及应用.重庆:重庆大学出版社,2002

11 李海涛,邓樱等编著.MATLAB 6.1基础及应用技巧.北京:国防工业出版社,2002

12 韩利竹,王华编著.MATLAB电子仿真与应用.北京:国防工业出版社,2001